Apple macOS 13 Ventura: FileVault Assurance Activity Report

Version:	1.1
Date:	2023-11-29
Status:	RELEASED
Classification:	Public
Filename:	VID11348_SER_AAR_Apple_FileVault_v1.1
Product:	Apple macOS 13 Ventura: FileVault
Sponsor:	Apple Inc.
Evaluation Facility:	atsec information security corporation
Validation ID:	11348
Validation Body:	NIAP CCEVS
Author(s):	Alex Gong, Stephan Mueller, Valerio Magliozzi, Walker Riley, Joachim Vandersmissen
Quality Assurance:	King Ables

atsec information security corporation
4516 Seton Center Pkwy, Suite 250
Austin, TX 78759

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

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
4516 Seton Center Pkwy, Suite 250
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
1.0	2023-11-03	Alex Gong, Walker Riley, Stephan Mueller, Joachim Vandersmissen	First version	
1.1	2023-11-29	Alex Gong, Walker Riley, Stephan Mueller, Joachim Vandersmissen	Address ECR comments	

Table of Contents

Classification: Public

List of Tables

List of Figures

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 defined in the following:

- [FDE_AA]☝: collaborative Protection Profile for Full Drive Encryption - Authorization Acquisition, Version 2.0 + Errata 20190201, dated 2019-02-01
- [FDE_EE]☝: collaborative Protection Profile for Full Drive Encryption - Encryption Engine, Version 2.0 + Errata 20190201, dated 2019-02-01

This evaluation claims Exact Compliance with the above Protection Profiles.

The following scheme documents and interpretations have been considered:

- [CCEVS-LG]☝: "CCEVS LabGrams", version as of November 2023.
- [CCEVS-PL]☝: "CCEVS Scheme Policy Letters", version as of November 2023.
- [CCEVS-PUB]☝: "CCEVS Scheme Publications", version as of November 2023.
- [CCEVS-TD]☝: "Technical Decisions", version as of November 2023.

2 Evaluation Results

This Assurance Activity Report covers the Apple macOS 13 Ventura: FileVault evaluation which claims conformance to the following Protection Profiles:

- [FDE_AA]☝: collaborative Protection Profile for Full Drive Encryption - Authorization Acquisition. Version 2.0 + Errata 20190201
- [FDE_EE]☝: collaborative Protection Profile for Full Drive Encryption - Encryption Engine. Version 2.0 + Errata 20190201

This report describes the evaluation team's assessment of the assurance activities and their results.

2.1 CAVP Summary

The CAVP certificates contain several different SoCs and micro-architectures in the operational environment (OE). The relationship between the SoCs and micro-architectures used by the devices claimed in this evaluation are specified in the Security Target.

The following convention has been used in the tables of this appendix to identify the cryptographic modules.

DMA

- Apple DMA Storage Controller 2.0 [Hardware], and/or
- Apple DMA Storage Controller 1.0 [Hardware]

KRN

- Apple corecrypto Module 13.0 [Apple ARM, Kernel, Software, SL1], and/or
- Apple corecrypto Module 13.0 [Intel, Kernel, Software, SL1]

SEP

- SEP Hardware v2.0 in Apple silicon, and/or
- SEP Hardware v2.0 in Apple T2

SKS

- Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2]

USR

- Apple corecrypto Module 13.0 [Apple ARM, User, Software, SL1], and/or
- Apple corecrypto Module 13.0 [Intel, User, Software, SL1]

The T2 is marketed as Apple ARM technology and runs T2OS 13.

The DMA module cannot be tested through the CAVP, therefore a compliance test accepted by NIAP has been used for verifying the correctness of the algorithms implemented.

Table 1: Cryptographic algorithm table

SFR	Algorithm	Capabilities	Mod	Implementation	CAVP
FCS_COP.1(a) Signature verification	ECDSA SigVer [FIPS 186-4]	Curve: P-521 with SHA-512 (Apple silicon)	USR	vng_ltc	A3488
			KRN	vng_ltc	A3521
			SKS	vng_ltc	A4259
	RSA SigVer [FIPS 186-4]	Modulo: 4096 with SHA-256 PKCS 1.5 and PKCSPSS (Intel)	USR	c_avx2	A3506
			KRN	c_avx2	A3623
		Modulo: 4096 with SHA-256	SKS	vng_ltc	A4109

SFR	Algorithm	Capabilities	Mod	Implementation	CAVP
		PKCS 1.5 and PKCSPSS (T2)			
FCS_COP.1(b) Hash	SHS Byte-oriented mode [FIPS 186-4]	SHA2-512 (Apple silicon)	USR	vng_ltc	A3488
			KRN	vng_ltc	A3521
			SKS	vng_ltc	A4259
	SHS Byte-oriented mode [FIPS 186-4]	SHA2-256 (Intel)	USR	vng_intel	A3512
			KRN	vng_intel	A3628
		SHA2-256 (T2)	SKS	vng_neon	A4110
FCS_COP.1(c)/AA / FCS_COP.1(c)/EE Keyed hash	HMAC Byte-oriented mode [ISO/IEC 9797-2:2011]	HMAC-SHA2-256 (Apple silicon)	SKS	vng_neon	A4260
		HMAC-SHA2-256 (T2)	SKS	vng_neon	A4110
FCS_COP.1(d) Key wrapping	AES [FIPS 197]	KW 256 bit encrypt, decrypt (Apple silicon) [SP800-38F]	SKS	c_asm	A4254
		KW 256 bit encrypt, decrypt (T2) [SP800-38F]	SKS	c_asm	A4104
FCS_COP.1(f) Data encrypt/decrypt	AES [FIPS 197]	XTS 256 bit encrypt, decrypt (Apple silicon) [SP800-38F]	DMA	n/a	None (verified through compliance test accepted by NIAP)
		XTS 128 bit encrypt, decrypt (Intel/T2)	DMA	n/a	None (verified through compliance

SFR	Algorithm	Capabilities	Mod	Implementation	CAVP
		[SP800-38F]			test accepted by NIAP)
FCS_COP.1(g) Key encryption	AES [ISO/IEC 18033-3]	CBC 256 bit encrypt [ISO/IEC 10116]	SEP	M2 Max (skg)	A3496
				M2 Pro (skg)	A3496
				M2 (skg)	A3496
				M1 Ultra (skg)	A3496
				M1 Max (skg)	A3496
				M1 Pro (skg)	A3496
				M1 (skg)	A1469
				T2 (skg)	C330
		CBC 256 bit encrypt, decrypt [ISO/IEC 10116]	SKS	M2 Max (c_asm)	A4254
				M2 Pro (c_asm)	A4254
				M2 (c_asm)	A4254
				M1 Ultra (c_asm)	A4254
				M1 Max (c_asm)	A4254
				M1 Pro (c_asm)	A4254
				M1 (c_asm)	A4254
				T2 (c_asm)	A4104
FCS_RBG_EXT.1 Random bit generation	CTR_DRBG [SP800-90A]	AES-256	SEP	M2 Max (trng)	A3490
				M2 Pro (trng)	A3490
				M2 (trng)	A3490
				M1 Ultra (trng)	A3490
				M1 Max (trng)	A3490
				M1 Pro (trng)	A3490
				M1 (trng)	A1362
				T2	DRBG 2029

The following table shows the coverage of CAVP tests for the Intel processors used in the devices covered by this evaluation and specified in Table 8: Hardware platforms in the Security Traget. For those processor models not tested, the last column indicates the equivalent processor on which the CAVP tests were performed.

The equivalency argument for these processors is that Intel processors within the same micro architecture and generation have the same instruction set and implementation design. The microcode executed during the assembler code is common to all processors for the same micro architecture. For instance, Intel Xeon W-3223, W-3235, W-3235, W-3245, W-3265, W-3265M and W-3275M belong to the same Micro Architecture ("Cascade Lake") and processor Generation ("W"). All these processors are equivalent in terms of design and instruction sets and any of them can be used as representative of the rest. Thus, algorithm testing under the CAVP were performed considering this equivalency. The CAVP testing was performed in at least one processor for each micro architecture and processor generation.

Table 2: Coverage of CAVP certificates for Intel Processors

Processor	Gen	Micro Architecture	USR		KRN		Equivalent processor
			RSA SigVer	SHA-256	RSA SigVer	SHA-256	
Intel Xeon W-2140B	W	Skylake	A3506	A3512	A3623	A3628	Tested
Intel Xeon W-2150B	W	Skylake					Intel Xeon W-2140B
Intel Xeon W-2170B	W	Skylake					Intel Xeon W-2140B
Intel Xeon W-2190B	W	Skylake					Intel Xeon W-2140B
Intel Xeon W-3223	W	Cascade Lake	A3506	A3512	A3623	A3628	Tested
Intel Xeon W-3235	W	Cascade Lake					Intel Xeon W-3223
Intel Xeon W-3245	W	Cascade Lake					Intel Xeon W-3223
Intel Xeon W-3265	W	Cascade Lake					Intel Xeon W-3223
Intel Xeon W-3265M	W	Cascade Lake					Intel Xeon W-3223
Intel Xeon W-3275M	W	Cascade Lake					Intel Xeon W-3223
Intel Core i5-8210Y	8th	Amber Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i5-8257U	8th	Coffee Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i5-8259U	8th	Coffee Lake					Intel Core i5-8257U
Intel Core i5-8279U	8th	Coffee Lake					Intel Core i5-8257U
Intel Core i7-8557U	8th	Coffee Lake					Intel Core i5-8257U
Intel Core i7-8559U	8th	Coffee Lake					Intel Core i5-8257U
Intel Core i7-8569U	8th	Coffee Lake					Intel Core i5-8257U
Intel Core i5-8500B	8th	Coffee Lake					Intel Core i7-8700B
Intel Core i7-8700B	8th	Coffee Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i7-8750H	8th	Coffee Lake					Intel Core i7-8700B
Intel Core i7-8850H	8th	Coffee Lake					Intel Core i7-8700B
Intel Core i9-8950HK	8th	Coffee Lake					Intel Core i7-8700B
Intel Core i7-9750H	9th	Coffee Lake					Intel Core i9-9880H
Intel Core i9-9880H	9th	Coffee Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i9-9880HK	9th	Coffee Lake					Intel Core i9-9880H
Intel Core i5-10500	10th	Comet Lake					Intel Core i7-10700K
Intel Core i5-10600	10th	Comet Lake					Intel Core i7-10700K
Intel Core i7-10700K	10th	Comet Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i9-10910	10th	Comet Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i5-1030NG7	10th	Ice Lake					Intel Core i7-1060NG7

Processor	Gen	Micro Architecture	USR		KRN		Equivalent processor
			RSA SigVer	SHA-256	RSA SigVer	SHA-256	
Intel Core i5-1038NG7	10th	Ice Lake					Intel Core i7-1060NG7
Intel Core i7-1060NG7	10th	Ice Lake	A3506	A3512	A3623	A3628	Tested
Intel Core i7-1068NG7	10th	Ice Lake					Intel Core i7-1060NG7

2.2 Security Functional Requirements

2.2.1 Cryptographic support (FCS)

2.2.1.1 Authorization Factor Acquisition (FCS_AFA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.1-ASE-01

> *The evaluator shall first examine the TSS to ensure that the authorization factors specified in the ST are described. For password-based factors the examination of the TSS section is performed as part of FCS_PCC_EXT.1 Evaluation Activities. Additionally in this case, the evaluator shall verify that the operational guidance discusses the characteristics of external authorization factors (e.g., how the authorization factor must be generated; format(s) or standards that the authorization factor must meet) that are able to be used by the TOE.*
>
> *If other authorization factors are specified, then for each factor, the TSS specifies how the factors are input into the TOE.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_AFA_EXT.1 which is covered in conjunction to FCS_KDF_EXT.1 and FCS_PCC_EXT.1. The TOE uses as Authorization Factor Acquisition a "password-based derivation function" (PBKDF2) as defined in SP800-132.

On both Apple silicon and "Intel with T2" Macs, the TOE supports password authentication factor. Passwords of up to 255 characters are supported and can be comprised of any combination of uppercase characters, lowercase characters, numbers, and any other 8-bit special character.

[CCGuide]☝ section 3.3 "User Accounts" provides instructions to set up the user's initial password which may be up to 255 characters in length and composed of printable ASCII characters (i.e., character codes 0x20 to 0x7E inclusive). This section also provides instructions to change the password.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.1-AGD-01

> *The evaluator shall verify that the AGD guidance includes instructions for all of the authorization factors. The AGD will discuss the characteristics of external authorization factors (e.g., how the authorization factor is generated; format(s) or standards that the authorization factor must meet, configuration of the TPM device used) that are able to be used by the TOE.*

Summary

Section 4 of [CCGuide] states the user account password is the authentication factor used to unlock the disk. Format and content of the password is described in secton 3.3.1, "Add Users."

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.1-ATE-01

> *The password authorization factor is tested in FCS_PCC_EXT.1.*
>
> *The evaluator shall also perform the following tests:*
>
> *Test 1 (conditional): If there is more than one authorization factor, ensure that failure to supply a required authorization factor does not result in access to the decrypted plaintext data.*

Summary

The password authorization factor is used to derive a cryptographic key (KEK), which is used to unwrap the data encryption keys (DEKs). This unwrapping procedure completes successfully if and only if the correct password is entered.

Using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore and verified that failure to supply the required authorization factor (password) does not result in access to the DEKs. As these DEKs are used to encrypt the plaintext data, the decrypted plaintext data is only accessible if the DEKs are available. In other words, the correct password must be supplied to access the decrypted plaintext data.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.1-AKM-01

> *The evaluator shall examine the Key Management Description to confirm that the initial authorization factors (submasks) directly contribute to the unwrapping of the BEV.*
>
> *The evaluator shall verify the KMD describes how a submask is produced from the authorization factor (including any associated standards to which this process might conform), and verification is performed to ensure the length of the submask meets the required size (as specified in this requirement).*

Summary

It is pointed out in Section 1.5.1 *Physical Boundary* of [ST] that the TOE includes both hardware and software running on two types of Apple Mac computers:

- Apple silicon Macs: Mac computers based on Apple silicon;
- "Intel with T2" Macs: Mac computers based on Intel CPU with Apple T2 security chip.

As mentioned in Section 2.1.1 *Key Chain from Password to BEV* of [KMD], the TOE supports the password authentication factor on both Apple silicon and "Intel with T2" Macs.

Section 2.1.1 of [KMD] describes the process of producing the submask from the user's password. On both Apple silicon and "Intel with T2" Macs, the TOE performs one iteration of PBKDF2 operation on the password to obtain a 256-bit key. The PBKDF2 is based on HMAC-SHA-256 algorithm and meets the standard of NIST SP 800-132. The 128-bit input salt value to the PBKDF2 is generated by the TRNG of the Secure Enclave when the password is set.

Next, this 256-bit output of PBKDF2 is repetitively encrypted with the hardware AES-CBC implementation using the 256-bit Secure Enclave UID as the key. This iterative encryption is performed as often as needed to reach a duration between 100 and 150 ms, with a minimum of 50,000 iterations. The 256-bit ciphertext of the last round of AES-CBC encryption forms the submask. This 256-bit submask is used as the Border Encryption Value (BEV) in the TOE.

Section 2.1.1 of [KMD]☝ also provides a diagram illustrating the process of producing the submask from the user's password. The text description and diagram demonstrate that the user's password directly contributes to producing the BEV.

The evaluator verified that the same submask size, 256 bits, is specified in the TSS for FCS_AFA_EXT.1 in Section 7 *TOE Summary Specification* of [ST]☝.

2.2.1.2 Timing of Authorization Factor Acquisition (FCS_AFA_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.2-ASE-01

> *The evaluator shall examine the TSS for a description of authorization factors and which of the factors are used to gain access to user data after the TOE entered a Compliant power saving state. The TSS is inspected to ensure it describes that each authorization factor satisfies the requirements of FCS_AFA_EXT.1.1.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_AFA_EXT.2. On both Apple silicon and Intel T2 devices, to resume from a compliant power state, the user must reauthenticate to the TOE. The user can reauthenticate using username and password.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.2-AGD-01

> *The evaluator shall examine the guidance documentation for a description of authorization factors used to access plaintext data when resuming from a Compliant power saving state.*

Summary

Per section 4 of [CCGuide]☝, *Usage*, when the system boots (resumes from a Compliant power saving state), the user will be prompted to select their account and enter their password authentication factor to unlock the disk.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.2-ATE-01

> *The evaluator shall perform the following test:*
>
> - *Enter the TOE into a Compliant power saving state*
> - *Force the TOE to resume from a Compliant power saving state*
> - *Release an invalid authorization factor and verify that access to decrypted plaintext data is denied*
> - *Release a valid authorization factor and verify that access to decrypted plaintext data is granted.*

Summary

The password authorization factor is used to derive a cryptographic key (KEK), which is used to unwrap the data encryption keys (DEKs). This unwrapping procedure completes successfully if and only if the correct password is entered.

The evaluator rebooted the device, causing it to enter and resume from the compliant G2(S5) (power off) power saving state. Then, using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore and verified that failure to supply the required authorization factor (password) does not result in access to the DEKs. As these DEKs are used to encrypt the plaintext data, the decrypted plaintext data is only accessible if the DEKs are available. In other words, the correct password must be supplied to access the decrypted plaintext data.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_AFA_EXT.2-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.3 Cryptographic Key Generation (Symmetric Keys) (FCS_CKM.1(b))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.1-B-ASE-01

> *The evaluator shall review the TSS to determine that a symmetric key is supported by the product, that the TSS includes a description of the protection provided by the product for this key. The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM.1(b). It states the following:

On both Apple silicon and "Intel with T2" Macs, the TOE generates intermediate symmetric keys of 256 bits using the random bit generator specified in FCS_RBG_EXT.1. The keys are generated by the Secure Enclave which invokes internally the TRNG to obtain random bits from the SP800-90A DRBG.

The TOE uses these intermediate symmetric keys to protect the key chain from the BEV to the DEK. Keys have 256 bits of security strength.

Assurance Activity AA-FDEEECPP-FCS_CKM.1-B-ASE-01

> *The evaluator shall review the TSS to determine that a symmetric key is supported by the product, that the TSS includes a description of the protection provided by the product for this key. The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM.1(b). The TOE generate symmetric keys for uses with AES-XTS according to FCS_COP.1(f) On the Apple silicon devices, the TOE generates 256-bit symmetric cryptographic keys (for AES-XTS-256) using the random bit generator specified in FCS_RBG_EXT.1. On the Intel T2 devices, the TOE generates 128-bit symmetric cryptographic keys (for AES-XTS-128) using the random bit generator specified in FCS_RBG_EXT.1.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.1-B-AGD-01

> *The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key size(s) for all uses specified by the AGD documentation and defined in this cPP.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Assurance Activity AA-FDEEECPP-FCS_CKM.1-B-AGD-01

> *The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key size(s) for all uses specified by the AGD documentation and defined in this cPP.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.1-B-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_CKM.1-B-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.1-B-AKM-01

> *If the TOE uses a symmetric key as part of the key chain, the KMD should detail how the symmetric key is used as part of the key chain.*

Summary

Section 2.1.1 *Key Chain from Password to BEV* of [KMD]☝ describes the key chain from the user's password to BEV in FDE AA component. The Secure Enclave UID is the only symmetric key used in the process of producing BEV from the user's password.

The TOE performs one iteration of PBKDF2 operation on the password. The 256-bit output of PBKDF2 is repetitively encrypted with the hardware AES-CBC implementation using the 256-bit Secure Enclave UID as the key. This iterative encryption is performed as often as needed to reach a duration between 100 and 150 ms, with a minimum of 50,000 iterations. The 256-bit ciphertext of the last round of AES-CBC encryption is defined as the BEV.

Assurance Activity AA-FDEEECPP-FCS_CKM.1-B-AKM-01

> *If the TOE uses a symmetric key as part of the key chain, the KMD should detail how the symmetric key is used as part of the key chain.*

Summary

Section 2.1 *Key Hierarchy* of [KMD] provides a detailed description of the key hierarchy from the user's password to BEV then to DEK. In particular, Section 2.1.2 *Key Chain from BEV to DEK* describes the key chain from the BEV to DEK, where the DEK is wrapped by AES-KW algorithm.

The DEK is different on Apple silicon Macs and "Intel with T2" Macs. On Apple silicon Macs, each file is encrypted with a separate 256-bit per-file key using the AES-XTS-256 cipher. The per-file key is defined as the DEK. On "Intel with T2" Macs, the data of a filesystem volume is encrypted with a single 256-bit volume key using the AES-XTS-128 cipher. The volume key is defined as the DEK.

When FileVault is enabled on Apple silicon Macs, the following symmetric keys are used as the decryption key for AES-KW algorithm to unwrap the DEK:

- BEV: 256-bit AES key derived from the user password and the Secure Enclave UID during the password authentication process.
- Media key: 256-bit AES key by the TRNG when the volume is created/initialized.
- Class C key (KEK): 256-bit AES key generated by the TRNG at the final stage of macOS installation or during the wipe operation replacing the old class C with a new key.

When FileVault is enabled on "Intel with T2" Macs, the following keys are used as the decryption key for AES-KW algorithm to unwrap the DEK:

- BEV: 256-bit AES key derived from the user password and the Secure Enclave UID during the password authentication process.
- Media key: 256-bit AES key by the TRNG when the volume is created/initialized.

2.2.1.4 Cryptographic Key Generation (Data Encryption Key) (FCS_CKM.1(c))

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.1-C-ASE-01

> *The evaluator shall examine the TSS to determine that it describes how the TOE obtains a DEK (either generating the DEK or receiving from the environment).*
>
> *If the TOE generates a DEK, the evaluator shall review the TSS to determine that it describes how the functionality described by FCS_RBG_EXT.1 is invoked. If the DEK is generated outside of the TOE, the evaluator checks to ensure that for each platform identified in the TOE the TSS, it describes the interface used by the TOE to invoke this functionality. The evaluator uses the description of the interface between the RBG and the TOE to determine that it requests a key greater than or equal to the required key sizes.*
>
> *If the TOE received the DEK from outside the host platform, then the evaluator shall examine the TSS to determine that the DEK is sent wrapped using the appropriate encryption algorithm.*

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM.1(c) which states the following:

- On both Apple silicon and "Intel with T2", the TOE generates a Data Encryption Key (DEK) of 256 bits using the random bit generator specified in FCS_RBG_EXT.1. The key is generated by the Secure Enclave which invokes internally the TRNG to obtain random bits from the SP800-90A DRBG. The DEK has a security strenght of 256 bits.
- On Apple silicon, the TOE uses the DEK to encrypt and decrypt data using AES-XTS-256 as described in TSS for FCS_COP.1(f). The DMA storage controller derives a 256-bit tweak and a 256-bit cipher key from this DEK.
- On Intel with T2, the TOE uses the DEK to encrypt and decrypt data using AES-XTS-128 as described in TSS for FCS_COP.1(f). The DMA storage controller splits the DEK into a 128-bit tweak and a 128-bit cipher key.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.1-C-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.1-C-ATE-01

> *The evaluator shall perform the following tests:*
>
> *Test 1: The evaluator shall configure the TOE to ensure the functionality of all selections.*

Summary

Using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after a factory reset. The evaluator verified that the contents of the key store are different, implying that new cryptographic keys were generated during factory reset, using the module's RBG.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.1-C-AKM-01

> *If the TOE received the DEK from outside the host platform, then the evaluator shall verify that the KMD describes how the TOE unwraps the DEK.*

Summary

Per Section 2.6 *Importing DEK into the TOE* of [KMD]☝, all symmetric cryptographic keys, including the DEK, are generated internal to the TOE using the TRNG of the Secure Enclave. The evaluator determined that the TOE does not receive the DEK from outside the host platform.

This work unit is not applicable and therefore considered to be satisfied.

2.2.1.5 Cryptographic Key Destruction (Power Management) (FCS_CKM.4(a)/AA)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-A-ASE-01

> *The evaluator shall verify the TSS provides a high level description of how keys stored in volatile memory are destroyed. The valuator to verify that TSS outlines:*
>
> - *if and when the TSF or the Operational Environment is used to destroy keys from volatile memory;*
> - *if and how memory locations for (temporary) keys are tracked;*
> - *details of the interface used for key erasure when relying on the OE for memory clearing.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM.4(a)/AA. The TOE will destroy all key material, BEV, and authentication factors stored in plaintext when transitioning to a Compliant power saving state as defined by and FPT_PWR_EXT.1/AA. The TOE erases cryptographic keys and key material from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-A-ASE-01

> *The evaluator shall verify the TSS provides a high level description of how keys stored in volatile memory are destroyed. The valuator to verify that TSS outlines: – if and when the TSF or the Operational Environment is used to destroy keys from volatile memory; – if and how memory locations for (temporary) keys are tracked; – details of the interface used for key erasure when relying on the OE for memory clearing.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM.4(a)/EE The TOE will destroy all key material, BEV, and authentication factors stored in plaintext when transitioning to a Compliant power saving state as defined by and FPT_PWR_EXT.1/EE. The TOE erases cryptographic keys and key material from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-A-AGD-01

> *The evaluator shall check the guidance documentation if the TOE depends on the Operational Environment for memory clearing and how that is achieved.*

Summary

Section 3.3.4 of [CCGuide]☝ states that the TOE does not depend on the Operational Environment for clearing memory.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-A-AGD-01

> *The evaluator shall check the guidance documentation if the TOE depends on the Operational Environment for memory clearing and how that is achieved.*

Summary

Section 3.3.4 of [CCGuide]☝ states that the TOE does not depend on the Operational Environment for clearing memory.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-A-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-A-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-A-AKM-01

> *The evaluator shall check to ensure the KMD lists each type of key, its origin, possible memory locations in volatile memory.*

Summary

The TOE consists of both FDE AA and FDE EE components. Section 2.2 *Cryptographic Keys* of [KMD]☝ describes the cryptographic keys used in the TOE. In particular, Subsections 2.2.2 and 2.2.3 describe each type of key used on Apple silicon Macs and "Intel with T2" Macs, respectively; Subsection 2.2.4 provides the information on how each type of key is generated and where that key is stored in volatile memory.

Based on the [KMD]☝, the evaluator compiled two tables as shown below, listing the following information of the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively:

- The origin of key;
- The location of key in volatile memory.

The term "SKS RAM" in the following tables denotes a DRAM memory region in the Operational Environment (TOE device) which is dedicated to the Secure Enclave.

Table 3: Origin and volatile memory location of keys on Apple silicon Macs

Key	Origin and Volatile memory location
Secure Enclave Unique ID (UID)	• Generated by the TRNG of the Secure Enclave and fused into Secure Enclave ROM at manufacture. • Volatile memory: Not stored there.
Root Encryption Key (REK)	• Derived from the user password and Secure Enclave UID during the password authentication process. • Volatile memory: Stored in SKS RAM.
Media key	• Generated by the TRNG when the volume is created/initialized. • Volatile memory: Stored in SKS RAM.
Per-file key	• Generated by the TRNG during the creation of a new file.

Key	Origin and Volatile memory location
	• Volatile memory: Stored in SKS RAM and in the DMA storage controller component.
Class C key	• Generated by the TRNG at the final stage of macOS installation or during the wipe operation replacing the old class C with a new key. • Volatile memory: Stored in SKS RAM.
DMA storage controller key	• Generated by the TRNG during each boot. • Volatile memory: Stored in SKS RAM and in the hardware register of the DMA storage controller.

Table 4: Origin and volatile memory location of keys on "Intel with T2" Macs

Key	Origin and Volatile memory location
Secure Enclave Unique ID (UID)	• Generated by the TRNG of the Secure Enclave and fused into Secure Enclave ROM at manufacture. • Volatile memory: Not stored there.
Root Encryption Key (REK)	• Derived from the user password and Secure Enclave UID during the password authentication process. • Volatile memory: Stored in SKS RAM.
Media key	• Generated by the TRNG when the volume is created/initialized. • Volatile memory: Stored in SKS RAM.
Volume key	• Generated by the TRNG during the creation of the filesystem volume. • Volatile memory: Stored in SKS RAM and in the DMA storage controller.
DMA storage controller key	• Generated by the TRNG during each boot. • Volatile memory: Stored in SKS RAM and in the hardware register of the DMA storage controller.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-A-AKM-01

The evaluator shall check to ensure the KMD lists each type of key, its origin, possible memory locations in volatile memory.

Summary

The TOE consists of both FDE AA and FDE EE components. The assurance activity for FCS_CKM.4(a) is the same on both FDE AA and FDE EE components.

Please see Table 3 and Table 4 for the provided information, which covers both FDE AA and FDE EE components.

2.2.1.6 Cryptographic Key Destruction (TOE-Controlled Hardware) (FCS_CKM.4(b))

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.4-B-ASE-01

(Key Management Description may be used if necessary details describe proprietary information)

The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.

The evaluator shall check to ensure the TSS lists each type of key that is stored, and identifies the memory type where key material is stored. When listing the type of memory employed, the TSS will list each type of memory selected in the FCS_CKM.4.1 SFR, as well as any memory types that employ a different memory controller or storage algorithm. For example, if a TOE uses NOR flash and NAND flash, both types are to be listed.

The evaluator shall examine the TSS to ensure it describes the method that is used by the memory controller to write and read memory from each type of memory listed. The purpose here is to provide a description of how the memory controller works so one can determine exactly how keys are written to memory. The description would include how the data is written to and read from memory (e.g., block level, cell level), mechanisms for copies of the key that could potentially exist (e.g., a copy with parity bits, a copy without parity bits, any mechanisms that are used for redundancy).

The evaluator shall examine the TSS to ensure it describes the destruction procedure for each key that has been identified. If different types of memory are used to store the key(s), the evaluator shall check to ensure that the TSS identifies the destruction procedure for each memory type where keys are stored (e.g., key X stored in flash memory is destroyed by overwriting once with zeros, key X' stored in EEPROM is destroyed by a overwrite consisting of a pseudo random pattern — the EEPROM used in the TOE uses a wear-leveling scheme as described).

If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.

The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement.

Upon completion of the TSS examination, the evaluator understands how all the keys (and potential copies) are destroyed.

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM.4(b). The TOE erases cryptographic keys and key material from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory. The TOE erases cryptographic keys and key material from non-volatile memory by performing a single overwrite of zeroes. The TOE leverages DRAM for volatile memory. Keys are stored in volatile memory while being used for their specific operation. Except for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed. The SEP performs the wrapping of keys, which are then sent to the memory controller for storage. The memory controller takes the block of data and the memory location provided by the SEP and stores the data in memory.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.4-B-AGD-01

There are a variety of concerns that may prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS and any other relevant Required Supplementary Information. The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-leveling and garbage collection. This may create additional copies of the key that are logically inaccessible but persist physically. In this case, it is assumed the drive supports the TRIM command and implements garbage collection to destroy these persistent copies when not actively engaged in other tasks.

Drive vendors implement garbage collection in a variety of different ways, as such there is a variable amount of time until data is truly removed from these solutions. There is a risk that data may persist for a longer amount of time if it is contained in a block with other data not ready for erasure. It is assumed the operating system and file system of the OE support TRIM, instructing the non-volatile memory to erase copies via garbage collection upon their deletion.

It is assumed that if a RAID array is being used, only set-ups that support TRIM are utilized. It is assumed if the drive is connected via PCI-Express, the operating system supports TRIM over that channel. It is assumed the drive is healthy and contains minimal corrupted data and will be end of life before a significant amount of damage to drive health occurs, it is assumed there is a risk small amounts of potentially recoverable data may remain in damaged areas of the drive.

Finally, it is assumed the keys are not stored using a method that would be inaccessible to TRIM, such as being contained in a file less than 982 bytes which would be completely contained in the master file table.

For destruction on wear-leveled memory, if a time period is required before is processed destruction the ST author shall provide an estimated range.

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that the TOE hardware consists of the Apple silicon SoC or Apple T2 Security Chip, which are custom silicon for Mac computers. The use of custom silicon ensures the key destruction requirements are always met and are not delayed by different hardware layers.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.4-B-ATE-01

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.

For destruction on wear-leveled memory, if a time period is required before is evaluator shall wait that amount of time after clearing the key in tests 2 and 3.

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 miniscule 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

Test 1: using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after a factory reset. The evaluator verified that the contents of the key store are different, implying that new cryptographic keys were generated during factory reset and the old cryptographic keys were destroyed.

Test 2: the procedure described in Test 1 covers both volatile and non-volatile keys.

Test 3: the procedure described in Test 1 covers both volatile and non-volatile keys.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM.4-B-AKM-01

(Key Management Description may be used if necessary details describe proprietary information)

The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.

The evaluator shall check to ensure the TSS lists each type of key that is stored, and identifies the memory type where key material is stored. When listing the type of memory employed, the TSS will list each type of memory selected in the FCS_CKM.4.1 SFR, as well as any memory types that employ a different memory controller or storage algorithm. For example, if a TOE uses NOR flash and NAND flash, both types are to be listed.

The evaluator shall examine the TSS to ensure it describes the method that is used by the memory controller to write and read memory from each type of memory listed. The purpose here is to provide a description of how the memory controller works so one can determine exactly how keys are written to memory. The description would include how the data is written to and read from memory (e.g., block level, cell level), mechanisms for copies of the key that could potentially exist (e.g., a copy with parity bits, a copy without parity bits, any mechanisms that are used for redundancy).

The evaluator shall examine the TSS to ensure it describes the destruction procedure for each key that has been identified. If different types of memory are used to store the key(s), the evaluator shall check to ensure that the TSS identifies the destruction procedure for each memory type where keys are stored (e.g., key X stored in flash memory is destroyed by overwriting once with zeros, key X' stored in EEPROM is destroyed by a overwrite consisting of a pseudo random pattern — the EEPROM used in the TOE uses a wear-leveling scheme as described).

If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.

The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement.

Upon completion of the TSS examination, the evaluator understands how all the keys (and potential copies) are destroyed.

Summary

The evaluator examined the TSS for FCS_CKM.4(b) in Section 7 *TOE Summary Specification* of [ST]☝, and verified that the TSS provides the required information on how the TOE's EE component meets this SFR. Therefore, Key Management Description is not needed for this SFR.

This work unit is not applicable and therefore considered to be satisfied.

2.2.1.7 Cryptographic Key Destruction (Software TOE, 3rd Party Storage) (FCS_CKM.4(d))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-D-ASE-01

> *(Key Management Description may be used if necessary details describe proprietary information)*
>
> *The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.*
>
> *The evaluator shall check to ensure the TSS lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).*
>
> *The evaluator examines the interface description for each different media type to ensure that the interface supports the selection(s) and description in the TSS.*
>
> *The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement. If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM.4(d). The TOE erases cryptographic keys from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory. The TOE erases cryptographic keys and key material from non-volatile memory by performing a single overwrite of zeroes. The TOE leverages DRAM for volatile memory. Keys are stored in volatile memory while being used for their specific operation. Except for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-D-ASE-01

> *(Key Management Description may be used if necessary details describe proprietary information)*
>
> *The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.*
>
> *The evaluator shall check to ensure the TSS lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).*
>
> *The evaluator examines the interface description for each different media type to ensure that the interface supports the selection(s) and description in the TSS.*
>
> *The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement. If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM.4(d). The TOE erases cryptographic keys from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory. The TOE erases cryptographic keys and key material from non-volatile memory by performing a single overwrite of zeroes. The TOE leverages DRAM for volatile memory. Keys are stored in volatile memory while being used for their specific operation. Except for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-D-AGD-01

> *There are a variety of concerns that may prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS and any other relevant Required Supplementary Information. The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.*
>
> *For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-leveling and garbage collection. This may create additional copies of the key that are logically inaccessible but persist physically. In this case, it is assumed the drive supports the TRIM command and implements garbage collection to destroy these persistent copies when not actively engaged in other tasks.*
>
> *Drive vendors implement garbage collection in a variety of different ways, as such there is a variable amount of time until data is truly removed from these solutions. There is a risk that data may persist for a longer amount of time if it is contained in a block with other data not ready for erasure. It is assumed the operating system and file system of the OE support TRIM, instructing the non-volatile memory to erase copies via garbage collection upon their deletion.*
>
> *It is assumed that if a RAID array is being used, only set-ups that support TRIM are utilized. It is assumed if the drive is connected via PCI-Express, the operating system supports TRIM over that channel. It is assumed the drive is healthy and contains minimal corrupted data and will be end of life before a significant amount of damage to drive health occurs, it is assumed there is a risk small amounts of potentially recoverable data may remain in damaged areas of the drive.*
>
> *Finally, it is assumed the keys are not stored using a method that would be inaccessible to TRIM, such as being contained in a file less than 982 bytes which would be completely contained in the master file table.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that the TOE hardware consists of the Apple silicon SoC or Apple T2 Security Chip, which are custom silicon for Mac computers. The use of custom silicon ensures the key destruction requirements are always met and are not delayed by different hardware layers.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-D-AGD-01

> *There are a variety of concerns that may prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS and any other relevant Required Supplementary Information. The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.*
>
> *For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-leveling and garbage collection. This may create additional copies of the key that are logically inaccessible but persist physically. In this case, it is assumed the drive supports the TRIM command and implements garbage collection to destroy these persistent copies when not actively engaged in other tasks.*
>
> *Drive vendors implement garbage collection in a variety of different ways, as such there is a variable amount of time until data is truly removed from these solutions. There is a risk that data may persist for a longer amount of time if it is contained in a block with other data not ready for erasure. It is assumed the operating system and file system of the OE support TRIM, instructing the non-volatile memory to erase copies via garbage collection upon their deletion.*

It is assumed that if a RAID array is being used, only set-ups that support TRIM are utilized. It is assumed if the drive is connected via PCI-Express, the operating system supports TRIM over that channel. It is assumed the drive is healthy and contains minimal corrupted data and will be end of life before a significant amount of damage to drive health occurs, it is assumed there is a risk small amounts of potentially recoverable data may remain in damaged areas of the drive.

Finally, it is assumed the keys are not stored using a method that would be inaccessible to TRIM, such as being contained in a file less than 982 bytes which would be completely contained in the master file table.

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that the TOE hardware consists of the Apple silicon SoC or Apple T2 Security Chip, which are custom silicon for Mac computers. The use of custom silicon ensures the key destruction requirements are always met and are not delayed by different hardware layers.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-D-ATE-01

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 miniscule 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.

The following tests apply only for the selection of “logically addresses the storage location…”, since the TOE in this instance has more visibility into what is happening within the underlying platform (e.g., a logical view of the media). For the selection of “instructs the underlying platform…”, the TOE has no visibility into the inner workings and completely relies on the underlying platform, so there is no reason to test the TOE beyond test 1.

For selection the selection of "logically addresses the storage location…", the following tests are used to determine the TOE is able to request the platform to overwrite the key with a TOE supplied pattern.

Test 2: Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use a tool that provides a logical view of the media (e.g., MBR file system):

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 logical view that 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 Use Case 1 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 a tool that provides a logical view of the media:

1. *Record the logical 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 logical 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

Test 1: Using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after a factory reset. The evaluator verified that the contents of the key store are different, implying that new cryptographic keys were generated during factory reset and the old cryptographic keys were destroyed.

Test 2: the procedure described in Test 1 covers both volatile and non-volatile keys.

Test 3: the procedure described in Test 1 covers both volatile and non-volatile keys.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-D-ATE-01

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 miniscule 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.

The following tests apply only to selection a), since the TOE in this instance has more visibility into what is happening within the underlying platform (e.g., a logical view of the media). In selection b), the TOE has no visibility into the inner workings and completely relies on the underlying platform, so there is no reason to test the TOE beyond test 1.

For selection a), the following tests are used to determine the TOE is able to request the platform to overwrite the key with a TOE supplied pattern.

Test 2: Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator shall use a tool that provides a logical view of the media (e.g., MBR file system):

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 logical view that 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 Use Case 1 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 a tool that provides a logical view of the media:

1. *Record the logical 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 logical 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

Test 1: using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after a factory reset. The evaluator verified that the contents of the key store are different, implying that new cryptographic keys were generated during factory reset and the old cryptographic keys were destroyed.

Test 2: this test is N/A (Not Applicable).

Test 3: this test is N/A (Not Applicable).

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM.4-D-AKM-01

> *(Key Management Description may be used if necessary details describe proprietary information)*
>
> *The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.*
>
> *The evaluator shall check to ensure the TSS lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).*
>
> *The evaluator examines the interface description for each different media type to ensure that the interface supports the selection(s) and description in the TSS.*
>
> *The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement. If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.*

Summary

The evaluator examined the TSS for FCS_CKM.4(d) in Section 7 *TOE Summary Specification* of [ST], and verified that the TSS provides the required information on how the TOE's AA component meets this SFR. Therefore, Key Management Description is not needed for this SFR.

This work unit is not applicable and therefore considered to be satisfied.

Assurance Activity AA-FDEEECPP-FCS_CKM.4-D-AKM-01

> *(Key Management Description may be used if necessary details describe proprietary information)*
>
> *The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.*
>
> *The evaluator shall check to ensure the TSS lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).*
>
> *The evaluator examines the interface description for each different media type to ensure that the interface supports the selection(s) and description in the TSS.*
>
> *The evaluator shall check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement. If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator shall verify that the pattern does not contain any CSPs.*

Summary

FCS_CKM.4(d) is a selection-based SFR in the [FDE_EE] cPP and the [ST] does not select this SFR for the TOE's EE component. This work unit is not applicable and therefore considered to be satisfied.

2.2.1.8 Cryptographic Key and Key Material Destruction (Destruction Timing) (FCS_CKM_EXT.4(a))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-A-ASE-01

> *The evaluator shall verify the TSS provides a high level description of what it means for keys and key material to be no longer needed and when then should be expected to be destroyed.*

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM_EXT.4(a). The TOE's Keys are only stored in volatile memory when they are required to perform a specific cryptographic operation. Since the keys are being used by the SEP to perform the operation, the SEP tracks the memory location of the key until the operation is complete. Once the keys are no longer required, the key that was used to perform the specific operation is erased from volatile memory by performing a single overwrite of zeroes. The erase operation is performed by the SEP and is not configurable by a user. There are no circumstances that do not conform to the key destruction requirement (e.g., sudden unexpected power loss).

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-A-ASE-01

> *The evaluator shall verify the TSS provides a high level description of what it means for keys and key material to be no longer needed and when then should be expected to be destroyed.*

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM_EXT.4(a). The TOE's Keys are only stored in volatile memory when they are required to perform a specific cryptographic operation. Since the keys are being used by the SEP to perform the operation, the SEP tracks the memory location of the key until the operation is complete. Once the keys are no longer required, the key that was used to perform the specific operation is erased from volatile memory by performing a single overwrite of zeroes. The erase operation is performed by the SEP and is not configurable by a user. There are no circumstances that do not conform to the key destruction requirement (e.g., sudden unexpected power loss).

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-A-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-A-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-A-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-A-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-A-AKM-01

> *The evaluator shall verify the KMD includes a description of the areas where keys and key material reside and when the keys and key material are no longer needed.*
>
> *The evaluator shall verify the KMD includes a key lifecycle, that includes a description where key material reside, how the key material is used, how it is determined that keys and key material are no longer needed, and how the material is destroyed once it is not needed and that the documentation in the KMD follows FCS_CKM.4(a) for the destruction.*

Summary

Section 2.2 *Cryptographic Keys* of [KMD], which consists of four subsections, describes the cryptographic keys used in the TOE. The TOE consists of both FDE AA and FDE EE components.

Subsection 2.2.1 describes the key destruction methods employed in the TOE to erase cryptographic keys and key material from volatile memory and non-volatile memory, respectively. The key destruction methods described in the [KMD] are consistent with FCS_CKM.4(d) in Section 6.1 *TOE Security Functional Requirements* of [ST]. Moreover, FCS_CKM.4(a)/AA in Section 6.1 of [ST] specifies that the TOE's AA component uses the key destruction methods specified in FCS_CKM.4(d). Therefore, [KMD] follows FCS_CKM.4(a)/AA for the destruction. It is stated in the [KMD] that the erase operation is performed by the Secure Enclave. Therefore, there are no circumstances that do not conform to the key destruction requirement.

Subsections 2.2.2 and 2.2.3 discuss the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively. The purpose of the keys are specified and the ways to use the keys are explained.

Subsection 2.2.4 supplements additional information of the cryptographic keys, including where in volatile memory and non-volatile memory the keys are stored, as well as when the keys are generated and deleted.

Altogether those four subsections provide the complete information of key lifecycle: its size, its purpose, its origin, whether it's stored in volatile memory, whether it's stored in the plaintext or ciphertext form, whether it's destroyable, as well as when and how to destroy that key.

The following two tables summarize the aforementioned information of the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively.

Table 5: Cryptographic keys used on Apple silicon Macs

Key	Information
Secure Enclave Unique ID (UID)	• 256-bit AES key for protecting device-specific secrets. • Generated by the TRNG of the Secure Enclave and fused into Secure Enclave ROM at manufacture. • Non-Volatile memory: Stored in the Secure Enclave ROM in plaintext. • Volatile memory: Not stored there.
Root Encryption Key (REK)	• 256-bit AES key used for user authentication and used as the BEV. • Derived from the user password and Secure Enclave UID during the password authentication process. • Non-Volatile memory: Not stored there. • Volatile memory: Temporarily stored in SKS RAM in plaintext, deleted from volatile memory after completing the BEV validation and user authentication.
Media key	• 256-bit AES key designed to enable swift and secure deletion of data. • Generated by the TRNG when the volume is created/initialized. • Non-Volatile memory: Secure Enclave effaceable storage, wrapped with Secure Enclave UID. Media key is designed to be quickly erased on demand, the wiping of the media key renders encrypted data inaccessible. • Volatile memory: Stored in SKS RAM in plaintext. Deleted from volatile memory when the system transitions to a Compliant power saving state.
Per-file key	• 256-bit AES key used as the DEK to encrypt data-at-rest of a file. • Generated by the TRNG during the creation of a new file. • Non-Volatile memory: Stored in each file's metadata, wrapped with KEK (Class C key). The file's metadata is further wrapped with Media Key. • Volatile memory: Stored in SKS RAM and the DMA storage controller component in plaintext when it is needed to read/write files. Deleted from volatile memory when the file operation has finished.
Class C key	• 256-bit AES key used as KEK to protect the DEK when FileVault is enabled. • Generated by the TRNG at the final stage of macOS installation or during the wipe operation replacing the old class C with a new key. • Non-volatile memory: Wrapped with the corresponding user-specific REK and stored in that user's keybag. The user's keybag is managed by the rich OS. • Volatile memory: Stored in SKS RAM in plaintext. Deleted from volatile memory when the system transitions to a Compliant power saving state.
DMA storage controller key	• 256-bit AES key used for secure transfer of the DEK from the Secure Enclave to DMA AES engine via the rich OS. • Generated by the TRNG during each boot. • Non-Volatile memory: Not stored there.

Key	Information
	• Volatile memory: Stored in 2 places in volatile memory in plaintext, SKS RAM and the hardware register of the DMA storage controller. Deleted from volatile memory when the system transitions to a Compliant power saving state.

Table 6: Cryptographic keys used on "Intel with T2" Macs

Key	Information
Secure Enclave Unique ID (UID)	• 256-bit AES key for protecting device-specific secrets. • Generated by the TRNG of the Secure Enclave and fused into Secure Enclave ROM at manufacture. • Non-Volatile memory: Stored in the Secure Enclave ROM in plaintext. • Volatile memory: Not stored there.
Root Encryption Key (REK)	• 256-bit AES key used for user authentication and used as the BEV. • Derived from the user password and Secure Enclave UID during the password authentication process. • Non-Volatile memory: Not stored there. • Volatile memory: Temporarily stored in SKS RAM in plaintext, deleted from volatile memory after completing the BEV validation and user authentication.
Media key	• 256-bit AES key designed to enable swift and secure deletion of data. • Generated by the TRNG when the volume is created/initialized. • Non-Volatile memory: Secure Enclave effaceable storage, wrapped with Secure Enclave UID. Media key is designed to be quickly erased on demand, the wiping of the media key renders encrypted data inaccessible. • Volatile memory: Stored in SKS RAM in plaintext. Deleted from volatile memory when the system transitions to a Compliant power saving state.
Volume key	• 256-bit AES key used as the DEK to encrypt data-at-rest of a filesystem volume. • Generated by the TRNG during the creation of the filesystem volume. • Non-Volatile memory: Stored in the filesystem volume. It's wrapped using the REK and then Media key. • Volatile memory: Stored in SKS RAM and in the DMA storage controller in plaintext after user authentication. Deleted from volatile memory when the system transitions to a Compliant power saving state or when the volume is unmounted.
DMA storage controller key	• 256-bit AES key used for secure transfer of the DEK from the Secure Enclave to DMA AES engine. • Generated by the TRNG during each boot. • Non-Volatile memory: Not stored there. • Volatile memory: Stored in 2 places in volatile memory in plaintext, SKS RAM and the hardware register of the DMA storage controller. Deleted from volatile memory when the system transitions to a Compliant power saving state.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-A-AKM-01

The evaluator shall verify the KMD includes a description of the areas where keys and key material reside and when the keys and key material are no longer needed.

The evaluator shall verify the KMD includes a key lifecycle, that includes a description where key material reside, how the key material is used, how it is determined that keys and key material are no longer needed, and how the material is destroyed once it is not needed and that the documentation in the KMD follows FCS_CKM.4(a) for the destruction.

Summary

Section 6.1 *TOE Security Functional Requirements* of [ST]☝ includes the following three SFRs, which are about the key destruction methods used in the TOE's EE component:

- FCS_CKM.4(a)/EE Cryptographic Key Destruction (Power Management) - Encryption Engine;
- FCS_CKM.4(b) Cryptographic Key Destruction (TOE-Controlled Hardware);
- FCS_CKM_EXT.6 Cryptographic Key Destruction Types.

FCS_CKM.4(a)/EE specifies that the TOE's EE component uses the key destruction methods specified in FCS_CKM_EXT.6. FCS_CKM_EXT.6 further specifies that the TOE's EE component uses the key destruction methods specified in FCS_CKM.4(b).

Section 2.2 *Cryptographic Keys* of [KMD]☝, which consists of four subsections, describes the cryptographic keys used in the TOE. The TOE consists of both FDE AA and FDE EE components.

Subsection 2.2.1 describes the key destruction methods employed in the TOE to erase cryptographic keys and key material from volatile memory and non-volatile memory, respectively. The key destruction methods described in the [KMD]☝ are consistent with FCS_CKM.4(b) in Section 6.1 of [ST]☝. Therefore, [KMD]☝ follows FCS_CKM.4(a)/EE for the destruction.

Subsections 2.2.2 and 2.2.3 discuss the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively. The purpose of the keys are specified and the ways to use the keys are explained.

Subsection 2.2.4 supplements additional information of the cryptographic keys, including where in volatile memory and non-volatile memory the keys are stored, as well as when the keys are generated and deleted.

Altogether those four subsections describe the lifecycle of the keys used in the TOE. Please refer to Table 5 and Table 6 for the provided information, which covers both FDE AA and FDE EE components.

2.2.1.9 Cryptographic Key and Key Material Destruction (Power Management) (FCS_CKM_EXT.4(b))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-B-ASE-01

> *The evaluator shall verify the TSS provides a description of what keys and key material are destroyed when entering any Compliant power saving state.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM_EXT.4(b). The TOE will destroy all key material, BEV, and authentication factors stored in plaintext when transitioning to a Compliant power saving state G2(S5) (soft off).

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-B-ASE-01

> *The evaluator shall verify the TSS provides a description of what keys and key material are destroyed when entering any Compliant power saving state.*

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_CKM_EXT.4(b). The TOE will destroy all key material, BEV, and authentication factors stored in plaintext when transitioning to a Compliant power saving state G2(S5) (soft off).

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-B-AGD-01

> *The evaluator shall validate that guidance documentation contains clear warnings and information on conditions in which the TOE may end up in a non-Compliant power saving state indistinguishable from a Compliant power saving state. In that case it must contain mitigation instructions on what to do in such scenarios.*

Summary

Section 3.2 of [CCGuide], *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state.

The low-power sleep state can be disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-B-AGD-01

> *The evaluator shall validate that guidance documentation contains clear warnings and information on conditions in which the TOE may end up in a non-Compliant power saving state indistinguishable from a Compliant power saving state. In that case it must contain mitigation instructions on what to do in such scenarios.*

Summary

Section 3.2 of [CCGuide], *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state.

The low-power sleep state can be disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-B-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-B-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_CKM_EXT.4-B-AKM-01

> *The evaluator shall verify the KMD includes a description of the areas where keys and key material reside.*
>
> *The evaluator shall verify the KMD includes a key lifecycle that includes a description where key material resides, how the key material is used, and how the material is destroyed once it is not needed and that the documentation in the KMD follows FCS_CKM.4(d) for the destruction.*

Summary

Section 2.2 *Cryptographic Keys* of [KMD], which consists of four subsections, describes the cryptographic keys used in the TOE.

Subsection 2.2.1 describes the key destruction methods employed in the TOE to erase cryptographic keys and key material from volatile memory and non-volatile memory, respectively. The key destruction methods described in the [KMD] are consistent with FCS_CKM.4(d) specified in Section 6.1 *TOE Security Functional Requirements* of [ST]. It is stated in the [KMD] that the erase operation is performed by the Secure Enclave. Therefore, there are no circumstances that do not conform to the key destruction requirement.

Subsections 2.2.2 and 2.2.3 discuss the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively. The purpose of the keys are specified and the ways to use the keys are explained.

Subsection 2.2.4 supplements additional information of the cryptographic keys, including where in volatile memory and non-volatile memory the keys are stored, as well as when the keys are generated and deleted.

Altogether those four subsections describe the lifecycle of the keys used in the TOE, including the following information:

- The storage location of key;
- The usage of key;
- The destruction of key.

Please refer to Table 5 and Table 6 for details.

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.4-B-AKM-01

> *The evaluator shall verify the KMD includes a description of the areas where keys and key material reside.*
>
> *The evaluator shall verify the KMD includes a key lifecycle that includes a description where key material resides, how the key material is used, and how the material is destroyed once it is not needed and that the documentation in the KMD follows FCS_CKM_EXT.6 for the destruction.*

Summary

Section 2.2 *Cryptographic Keys* of [KMD], which consists of four subsections, describes the cryptographic keys used in the TOE. The TOE consists of both FDE AA and FDE EE components.

Subsection 2.2.1 describes the key destruction methods employed in the TOE to erase cryptographic keys and key material from volatile memory and non-volatile memory, respectively. The key destruction methods described in the [KMD] are consistent with FCS_CKM.4(b) in Section 6.1 *TOE Security Functional Requirements* of [ST]. FCS_CKM_EXT.6 in Section 6.1 of [ST] specifies that the TOE's EE component uses the key destruction methods specified in FCS_CKM.4(b). Therefore, [KMD] follows FCS_CKM.6 for the destruction.

Subsections 2.2.2 and 2.2.3 discuss the cryptographic keys used on Apple silicon Macs and "Intel with T2" Macs, respectively. The purpose of the keys are specified and the ways to use the keys are explained.

Subsection 2.2.4 supplements additional information of the cryptographic keys, including where in volatile memory and non-volatile memory the keys are stored, as well as when the keys are generated and deleted.

Altogether those four subsections describe the lifecycle of the keys used in the TOE, including the following information:

- The storage location of key;
- The usage of key;
- The destruction of key.

Please refer to Table 5 and Table 6 for the provided information, which covers both FDE AA and FDE EE components.

2.2.1.10 Cryptographic Key Destruction Types (FCS_CKM_EXT.6)

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.6-ASE-01

> *The evaluator shall examine the TOE's keychain in the TSS and verify all keys subject to destruction are destroyed according to one of the specified methods.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_CKM_EXT.6 which is covered in conjunction to FCS_CKM.4(a)/AA FCS_CKM.4(a)/EE FCS_CKM.4(b) FCS_CKM.4(d) FCS_CKM_EXT.4(a) FCS_CKM_EXT.4(b) FPT_KYP_EXT.1/AA FPT_KYP_EXT.1/EE. The TOE leverages DRAM for volatile memory. Keys are stored in volatile memory while being used for their specific operation. Except for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed. The TOE erases cryptographic keys and key material from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory. The TOE erases cryptographic keys and key material from non-volatile memory by performing a single overwrite of zeroes.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.6-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.6-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_CKM_EXT.6-AKM-01

> *The evaluator shall examine the TOE's keychain in the KMD and verify all keys subject to destruction are destroyed according to one of the specified methods.*

Summary

Section 2.2.1 *Storage and Destruction of Cryptographic keys* of [KMD]☝ describes the key destruction methods employed in the TOE:

- The TOE erases cryptographic keys and key material from volatile memory by performing a single overwrite of zeroes and/or by removal of power to the memory.
- The TOE erases cryptographic keys and key material from non-volatile memory by performing a single overwrite of zeroes.

The key destruction methods described in the [KMD]☝ are consistent with Section 7 *TOE Summary Specification* of [ST]☝.

2.2.1.11 Cryptographic Operation (Signature Verification) (FCS_COP.1(a))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-A-ASE-01

> *The evaluator shall check the TSS to ensure that it describes the overall flow of the signature verification. This should at least include identification of the format and general location (e.g., "firmware on the hard drive device" rather than "memory location 0x00007A4B") of the data to be used in verifying the digital signature; how the data received from the operational environment are brought on to the device; and any processing that is performed that is not part of the digital signature algorithm (for instance, checking of certificate revocation lists).*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(a). The TOE perform signature verification as part of the following features:

- Installing firmware/software updates
  - FPT_FUA_EXT.1 and FPT_TUD_EXT.1
- Secure Boot
  - FPT_TST_EXT.1

Installation and Secure Boot signature verification involves different TOE components in different layers of the TOE and, thus, use the user space, kernel space, and SKS corecrypto modules. Verification signature algorithm are provided for each platform:
On Apple silicon:

- Algorithm: ECDSA P-521 sigver according to FIPS PUB 186-4 standard.
- Used in modules:

    - Apple corecrypto Module 13.0 [Apple ARM, User, Software, SL1]
    - Apple corecrypto Module 13.0 [Apple ARM, Kernel, Software, SL1]
    - Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2]

Signatures are verified using ECDSA P-521 and SHA-512. The CA public key is embedded in the devices Boot ROM code during manufacturing. The TOE image is signed using this key's corresponding private key.
On Intel T2:

- Algorithm: RSA 4096 sigver according to IEEE 1619 standard
- Used in modules:
    - Apple corecrypto Module 13.0 [Intel, User, Software, SL1]
    - Apple corecrypto Module 13.0 [Intel, Kernel, Software, SL1]
    - Apple corecrypto Module 13.0 [Apple T2, Secure Key Store, Hardware, SL2]

signatures are verified using RSA 4096-bit and SHA-256. The CA public key is embedded in the SEP's Boot ROM code during manufacturing. The TOE image is signed using this key's corresponding private key.

Assurance Activity AA-FDEEECPP-FCS_COP.1-A-ASE-01

> *The evaluator shall check the TSS to ensure that it describes the overall flow of the signature verification. This should at least include identification of the format and general location (e.g., "firmware on the hard drive device" rather than “memory location 0x00007A4B") of the data to be used in verifying the digital signature; how the data received from the operational environment are brought on to the device; and any processing that is performed that is not part of the digital signature algorithm (for instance, checking of certificate revocation lists).*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(a). The TOE perform signature verification as part of the following features:

- Installing firmware/software updates
    - FPT_FUA_EXT.1 and FPT_TUD_EXT.1
- Secure Boot
    - FPT_TST_EXT.1

Installation and Secure Boot signature verification involves different TOE components in different layers of the TOE and, thus, use the user space, kernel space, and SKS corecrypto modules. Verification signature algorithm are provided for each platform:
On Apple silicon:

- Algorithm: ECDSA P-521 sigver according to FIPS PUB 186-4 standard.
- Used in modules:
    - Apple corecrypto Module 13.0 [Apple ARM, User, Software, SL1]
    - Apple corecrypto Module 13.0 [Apple ARM, Kernel, Software, SL1]
    - Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2]

Signatures are verified using ECDSA P-521 and SHA-512. The CA public key is embedded in the devices Boot ROM code during manufacturing. The TOE image is signed using this key's corresponding private key.
On Intel T2:

- Algorithm: RSA 4096 sigver according to IEEE 1619 standard
- Used in modules:
  - Apple corecrypto Module 13.0 [Intel, User, Software, SL1]
  - Apple corecrypto Module 13.0 [Intel, Kernel, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple T2, Secure Key Store, Hardware, SL2]

signatures are verified using RSA 4096-bit and SHA-256. The CA public key is embedded in the SEP's Boot ROM code during manufacturing. The TOE image is signed using this key's corresponding private key.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-A-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-A-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

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

Each section below contains the tests the evaluators must perform for each type of digital signature scheme. Based on the assignments and selections in the requirement, the evaluators choose the specific activities that correspond to those selections.

It should be noted that for the schemes given below, there are no key generation/domain parameter generation testing requirements. This is because it is not anticipated that this functionality would be needed in the end device, since the functionality is limited to checking digital signatures in delivered updates. This means that the domain parameters should have already been generated and encapsulated in the hard drive firmware or on-board non-volatile storage. If key generation/domain parameter generation is required, the evaluation and validation scheme must be consulted to ensure the correct specification of the required evaluation activities and any additional components.

The following tests are conditional based upon the selections made within the SFR.

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

ECDSA Algorithm Tests

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.

RSA Signature Algorithm Tests

Signature Verification Test
The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's authentic and unauthentic 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

This test is covered by CAVP certs A3488, A3521, A4259, A3506, A3623, and A4109.

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

Each section below contains the tests the evaluators must perform for each type of digital signature scheme. Based on the assignments and selections in the requirement, the evaluators choose the specific activities that correspond to those selections.

It should be noted that for the schemes given below, there are no key generation/domain parameter generation testing requirements. This is because it is not anticipated that this functionality would be needed in the end device, since the functionality is limited to checking digital signatures in delivered updates. This means that the domain parameters should have already been generated and encapsulated in the hard drive firmware or on-board non-volatile storage. If key generation/domain parameter generation is required, the evaluation and validation scheme must be consulted to ensure the correct specification of the required evaluation activities and any additional components.

The following tests are conditional based upon the selections made within the SFR.

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

ECDSA Algorithm Tests

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.

RSA Signature Algorithm Tests

Signature Verification Test The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's authentic and unauthentic 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

This test is covered by CAVP certs A3488, A3521, A4259, A3506, A3623, and A4109.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-A-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-A-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.12 Cryptographic Operation (Hash Algorithm) (FCS_COP.1(b))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-B-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.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(b). The TOE support hash functions, for each platform:
Apple Silicon:

- Algorithm: SHA-512 according to ISO/IEC 10118-3:2004 standard
- Used in modules:
  - Apple corecrypto Module 13.0 [Apple ARM, User, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple ARM, Kernel, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2]

On Intel T2:

- Algorithm: SHA-256 according to ISO/IEC 10118-3:2004 standard
- Used in modules:
  - Apple corecrypto Module 13.0 [Intel, User, Software, SL1]
  - Apple corecrypto Module 13.0 [Intel, Kernel, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple T2, Secure Key Store, Hardware, SL2]

Assurance Activity AA-FDEEECPP-FCS_COP.1-B-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.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(b). The TOE support hash functions, for each platform:
Apple Silicon:

- Algorithm: SHA-512 according to ISO/IEC 10118-3:2004 standard
- Used in modules:
  - Apple corecrypto Module 13.0 [Apple ARM, User, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple ARM, Kernel, Software, SL1]
  - Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2]

On Intel T2:

- Algorithm: SHA-256 according to ISO/IEC 10118-3:2004 standard
- Used in modules:

- Apple corecrypto Module 13.0 [Intel, User, Software, SL1]
- Apple corecrypto Module 13.0 [Intel, Kernel, Software, SL1]
- Apple corecrypto Module 13.0 [Apple T2, Secure Key Store, Hardware, SL2]

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-B-AGD-01

> *The evaluator checks the operational guidance documents to determine that any system configuration necessary to enable required hash size functionality is provided.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Assurance Activity AA-FDEEECPP-FCS_COP.1-B-AGD-01

> *The evaluator checks the operational guidance documents to determine that any system configuration necessary to enable required hash size functionality is provided.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Test Assurance Activities

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

> *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 test mode.*
>
> *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 cPP.*
>
> *<u>Short Messages Test Bit-oriented Mode</u>*
> *The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*
>
> *<u>Short Messages Test Byte-oriented Mode</u>*
> *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.*
>
> *<u>Selected Long Messages Test Bit-oriented Mode</u>*
> *The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. For SHA-256, the length of the i-th message is 512 + 99*i, where 1 *lt;= i <= m. For SHA-384 and SHA-512, the length of the i-th message is 1024 + 99*i, where 1 <= i <= m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Selected Long Messages Test Byte-oriented Mode
*The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. For SHA-256, the length of the i-th message is 512 + 8*99*i, where 1 <= i <= m/8. For SHA-384 and SHA-512, the length of the i-th message is 1024 + 8*99*i, where 1 <= i <= m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Pseudorandomly Generated Messages Test
This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of the NIST Secure Hash Algorithm Validation System (SHAVS) (https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation-Program/documents/shs/SHAVS.pdf). The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

Summary

This test is covered by CAVP certs A3488, A3521, A4259, A3512, A3628, and A4100.

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

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 test mode.

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 cPP.

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

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

*Selected Long Messages Test Bit-oriented Mode The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. For SHA-256, the length of the i-th message is 512 + 99*i, where 1 <= i <= m. For SHA-384 and SHA-512, the length of the i-th message is 1024 + 99*i, where 1 <= i <= m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

*Selected Long Messages Test Byte-oriented Mode The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. For SHA-256, the length of the i-th message is 512 + 8*99*i, where 1 <= i <= m/8. For SHA-384 and SHA-512, the length of the i-th message is 1024 + 8*99*i, where 1 <= i <= m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Pseudorandomly Generated Messages Test This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of the NIST Secure Hash Algorithm Validation System (SHAVS) (https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation- Program/documents/shs/SHAVS.pdf). The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

Summary

This test is covered by CAVP certs A3488, A3521, A4259, A3512, A3628, and A4100.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-B-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-B-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.13 Cryptographic Operation (Keyed Hash Algorithm) (FCS_COP.1(c))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-C-ASE-01

> *If HMAC was selected:*
>
> *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.*
>
> *If CMAC was selected:*
>
> *The evaluator shall examine the TSS to ensure that it specifies the following values used by the CMAC function: key length, block cipher used, block size (of the cipher), and output MAC length used.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(c)/AA. The TOE provides keyed-hash message auhthentication using HMAC-SHA-256 algorithm according to ISO/IEC 9797-2:2011 standard. On both Apple silicon and Intel T2 devices, the PBKDF2 uses the keyed hash algorithm HMAC-SHA-256 from the corecrypto SKS module as described in the FCS_PCC_EXT.1. The algorithm supports a key size of 256 bits.

Assurance Activity AA-FDEEECPP-FCS_COP.1-C-ASE-01

> *If HMAC was selected:*
>
> *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.*
>
> *If CMAC was selected:*
>
> *The evaluator shall examine the TSS to ensure that it specifies the following values used by the CMAC function: key length, block cipher used, block size (of the cipher), and output MAC length used.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(c)/EE. The TOE provides Keysed-hash message auhthentication using HMAC-SHA-256 algorithm according to ISO/IEC 9797-2:2011 standard. On both Apple silicon and Intel T2 devices, the PBKDF2 uses the keyed hash algorithm HMAC-SHA-256 from the corecrypto SKS module as described in the FCS_PCC_EXT.1. The algorithm supports a key size of 256 bits.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-C-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-C-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

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

If HMAC was selected:

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 using a known good implementation.

If CMAC was selected:

For each of the supported parameter sets, the evaluator shall compose at least 15 sets of test data. Each set shall consist of a key and message data. The test data shall include messages of different lengths, some with partial blocks as the last block and some with full blocks as the last block. The test data keys shall include cases for which subkey K1 is generated both with and without using the irreducible polynomial R_b, as well as cases for which subkey K2 is generated from K1 both with and without using the irreducible polynomial R_b. (The subkey generation and polynomial R_b are as defined in SP800-38E.) The evaluator shall have the TSF generate CMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating CMAC tags with the same key using a known good implementation.

Summary

This test is covered by CAVP certs A4260 and A4110.

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

If HMAC was selected:

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 using a known good implementation.

If CMAC was selected:

For each of the supported parameter sets, the evaluator shall compose at least 15 sets of test data. Each set shall consist of a key and message data. The test data shall include messages of different lengths, some with partial blocks as the last block and some with full blocks as the last block. The test data keys shall include cases for which

> *subkey K1 is generated both with and without using the irreducible polynomial R_b, as well as cases for which subkey K2 is generated from K1 both with and without using the irreducible polynomial R_b. (The subkey generation and polynomial R_b are as defined in SP800-38E.) The evaluator shall have the TSF generate CMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating CMAC tags with the same key using a known good implementation.*

Summary

This test is covered by CAVP certs A4260 and A4110.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-C-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-C-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.14 Cryptographic Operation (Key Wrapping) (FCS_COP.1(d))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-D-ASE-01

> *The evaluator shall verify the TSS includes a description of the key wrap function(s) and shall verify the key wrap uses an approved key wrap algorithm according to the appropriate specification.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_COP.1(d) which states the following that on both Apple silicon and "Intel with T2" Macs, the TOE performs key wrapping using the AES in KW mode according to [SP800-38F]☝. The TOE uses 256-bit keys for this algorithm. AES-KW is an authentication cipher that provides integrity: the decryption operation will only succeed when there is no authentication error. This ensures that the unwrapping operation is performed with the correct key.

Assurance Activity AA-FDEEECPP-FCS_COP.1-D-ASE-01

> *The evaluator shall verify the TSS includes a description of the key wrap function(s) and shall verify the key wrap uses an approved key wrap algorithm according to the appropriate specification.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_COP.1(d) which states the following that on both Apple silicon and "Intel with T2" Macs, the TOE performs key wrapping using the AES in KW mode according to [SP800-38F]☝. The TOE uses 256-bit keys for this algorithm. AES-KW is an authentication cipher that provides integrity: the decryption operation will only succeed when there is no authentication error. This ensures that the unwrapping operation is performed with the correct key.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-D-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-D-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

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

> *There are no test evaluation activities for this SFR.*

Summary

This test is covered by CAVP certs A4254 and A4104.

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

> *There are no test evaluation activities for this SFR.*

Summary

This test is covered by CAVP certs A4254 and A4104.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-D-AKM-01

> *The evaluator shall review the KMD to ensure that all keys are wrapped using the approved method and a description of when the key wrapping occurs.*

Summary

As mentioned in Section 2.2.4 *Additional Information of Keys* of [KMD]☝, the following keys are stored in the non-volatile memory in the wrapped form:

- Class C key: KEK on Apple silicon Macs when FileVault is enabled.
- Per-file key: DEK on Apple silicon Macs.
- Volume key: DEK on "Intel with T2" Macs.

FCS_COP.1(d) in Section 6.1 *TOE Security Functional Requirements* of [ST]☝ indicates that the TOE uses the AES-KW-256 key wrapping method meeting the standard of NIST SP 800-38F. Section 2.1.2 *Key Chain from BEV to DEK* of [KMD]☝ confirms the same AES-KW-256 key wrapping method is used in the TOE.

After being created, the aforementioned keys are wrapped and stored in the non-volatile memory. Section 2.2.4 *Additional Information of Keys* of [KMD]☝ provides the detail:

- Class C key is wrapped with the user's REK and stored in that user's keybag.
- Per-file key is wrapped with KEK and stored in the file's metadata. The file's metadata is further wrapped with Media Key.
- Volume key is wrapped using the REK and then Media key. It is stored in the filesystem volume.

In addition, Sections 2.2.2 and 2.2.3 of [KMD]☝ indicate that, while being transferred from the Secure Enclave to DMA storage controller, the DEK (the per-file key on Apple silicon Macs and the volume key on "Intel with T2" Macs) is wrapped with DMA storage controller key.

Assurance Activity AA-FDEEECPP-FCS_COP.1-D-AKM-01

> *The evaluator shall review the KMD to ensure that all keys are wrapped using the approved method and a description of when the key wrapping occurs.*

Summary

Per FCS_COP.1(d) in Section 6.1 *TOE Security Functional Requirements* of [ST]☝, the TOE adopts the same key wrapping methods for both FDE AA and FDE EE components.

As mentioned in Section 2.2.4 *Additional Information of Keys* of [KMD]☝, the following keys are stored in the non-volatile memory in the wrapped form:

- Class C key: KEK on Apple silicon Macs when FileVault is enabled.
- Per-file key: DEK on Apple silicon Macs.
- Volume key: DEK on "Intel with T2" Macs.

FCS_COP.1(d) in Section 6.1 *TOE Security Functional Requirements* of [ST]☝ indicates that the TOE uses the AES-KW-256 key wrapping method meeting the standard of NIST SP 800-38F. Section 2.1.2 *Key Chain from BEV to DEK* of [KMD]☝ confirms the same AES-KW-256 key wrapping method is used in the TOE.

After being created, the aforementioned keys are wrapped and stored in the non-volatile memory. Section 2.2.4 *Additional Information of Keys* of [KMD]☝ provides the detail:

- Class C key is wrapped with the user's REK and stored in that user's keybag.
- Per-file key is wrapped with KEK and stored in the file's metadata. The file's metadata is further wrapped with Media Key.
- Volume key is wrapped using the REK and then Media key. It is stored in the filesystem volume.

In addition, Sections 2.2.2 and 2.2.3 of [KMD]☝ indicate that, while being transferred from the Secure Enclave to DMA storage controller, the DEK (the per-file key on Apple silicon Macs and the volume key on "Intel with T2" Macs) is wrapped with DMA storage controller key.

2.2.1.15 Cryptographic Operation (AES Data Encryption/Decryption) (FCS_COP.1(f))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-F-ASE-01

> *The evaluator shall verify the TSS includes a description of the key size used for encryption and the mode used for encryption.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(f). The TOE uses AES-XTS for data encryption and decryption. The symmetric keys are generated as specified in FCS_CKM.1(b).
Apple Silicon:

- Algorithm: AES-XTS-256 according to IEEE 1619 standard
- Used in module:
    - Apple Storage Controller 2.0 Hardware

On Apple silicon devices, the TOE supports AES data encryption and decryption in XTS mode using two independent 256-bit keys. Each key contains 256 bits of entropy obtained from the TRNG.
Intel T2:

- Algorithm: AES-XTS-128 according to IEEE 1619 standard
- Used in module:
    - Apple Storage Controller 2.0 Hardware

On Intel T2 devices, the TOE supports AES data encryption and decryption in XTS mode using two independent 128-bit keys. Each key contains 128 bits of entropy obtained from the TRNG.

Assurance Activity AA-FDEEECPP-FCS_COP.1-F-ASE-01

> *The evaluator shall verify the TSS includes a description of the key size used for encryption and the mode used for encryption.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_COP.1(f). The TOE uses AES-XTS for data encryption and decryption. the symmetric keys are generate according to FCS_CKM.1(b). The used algorithm depends on the platform:

- **Apple silicon** Algorithm: AES-XTS-256 according to standard: IEEE 1619 for the Apple DMA controller 2.0 [Hardware] module. On Apple silicon devices, the TOE supports AES data encryption and decryption in XTS mode using two independent 256-bit keys. Each key contains 256 bits of entropy obtained from the TRNG.

- **Intel T2** Algorithm: AES-XTS-128 according to standard: IEEE 1619 for the Apple DMA controller 1.0 [Hardware]module. On "Intel with T2" devices, the TOE supports AES data encryption and decryption in XTS mode using two independent 128-bit keys. Each key contains 128 bits of entropy obtained from the TRNG.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-F-AGD-01

If multiple encryption modes are supported, the evaluator examines the guidance documentation to determine that the method of choosing a specific mode/key size by the end user is described.

Summary

This SFR is selection-based in [FDE_AA]☝ and not claimed in [ST]☝ from this PP, but from [FDE_EE]☝ and thus addressed in AA-FDEEECPP-FCS_COP.1-F-AGD-01. This work unit from [FDE_AA]☝ is therefore not applicable and considered implicitly satisfied.

Assurance Activity AA-FDEEECPP-FCS_COP.1-F-AGD-01

If multiple encryption modes are supported, the evaluator examines the guidance documentation to determine that the method of choosing a specific mode/key size by the end user is described.

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration choice of encryption modes or key sizes is necessary or available.

Test Assurance Activities

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

The following tests are conditional based upon the selections made in the SFR.

AES-CBC Tests

For the AES-CBC tests described below, the plaintext, ciphertext, and IV values shall consist of 128-bit blocks. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known-good implementation.

These tests are intended to be equivalent to those described in NIST's AES Algorithm Validation Suite (AESAVS) (http://csrc.nist.gov/groups/STM/cavp/documents/aes/AESAVS.pdf). Known answer values tailored to exercise the AES-CBC implementation can be obtained using NIST's CAVS Algorithm Validation Tool or from NIST's ACVP service for automated algorithm tests (acvp.nist.gov), when available. It is not recommended that evaluators use values obtained from static sources such as the example NIST's AES Known Answer Test Values from the AESAVS document, or use values not generated expressly to exercise the AES-CBC implementation.

AES-CBC Known Answer Tests

KAT-1 (GFSBox):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five different plaintext values for each selected key size 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.

To test the decrypt functionality of AES-CBC, the evaluator shall supply a set of five different ciphertext values for each selected key size and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using a key value of all zeros and an IV of all zeros.

KAT-2 (KeySBox):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five different key values for each selected key size 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.

To test the decrypt functionality of AES-CBC, the evaluator shall supply a set of five different key values for each selected key size and obtain the plaintext that results from AES-CBC decryption of an all-zeros ciphertext using the given key and an IV of all zeros.

KAT-3 (Variable Key):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of keys for each selected key size (as described below) and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using each key and an IV of all zeros.

Key i in each set shall have the leftmost i bits set to ones and the remaining bits to zeros, for values of i from 1 to the key size. The keys and corresponding ciphertext are listed in AESAVS, Appendix E.

To test the decrypt functionality of AES-CBC, the evaluator shall use the same keys as above to decrypt the ciphertext results from above. Each decryption should result in an all-zeros plaintext.

KAT-4 (Variable Text):

To test the encrypt functionality of AES-CBC, for each selected key size, the evaluator shall supply a set of 128-bit plaintext values (as described below) and obtain the ciphertext values that result from AES-CBC encryption of each plaintext value using a key of each size and IV consisting of all zeros.

Plaintext value i shall have the leftmost i bits set to ones and the remaining bits set to zeros, for values of i from 1 to 128. The plaintext values are listed in AESAVS, Appendix D.

To test the decrypt functionality of AES-CBC, for each selected key size, use the plaintext values from above as ciphertext input, and AES-CBC decrypt each ciphertext value using key of each size consisting of all zeros and an IV of all zeros.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting nine i-block messages for each selected key size, for 2 <= i <= 10. For each test, the evaluator shall supply a key, an IV, and a plaintext message of length i blocks, and encrypt the message using AES-CBC. The resulting ciphertext values shall be compared to the results of encrypting the plaintext messages using a known good implementation.

The evaluator shall test the decrypt functionality by decrypting nine i-block messages for each selected key size, for 2 <= i <= 10. For each test, the evaluator shall supply a key, an IV, and a ciphertext message of length i blocks, and decrypt the message using AES-CBC. The resulting plaintext values shall be compared to the results of decrypting the ciphertext messages using a known good implementation.

AES-CBC Monte Carlo Tests

The evaluator shall test the encrypt functionality for each selected key size using 100 3-tuples of pseudo-random values for plaintext, IVs, and keys.

The evaluator shall supply a single 3-tuple of pseudo-random values for each selected key size. This 3-tuple of

plaintext, IV, and key is provided as input to the below algorithm to generate the remaining 99 3-tuples, and to run each 3-tuple through 1000 iterations of AES-CBC encryption.

```
# Input: PT, IV, Key
Key[0] = Key
IV[0] = IV
PT[0] = PT

for i = 1 to 100 {
    Output Key[i], IV[i], PT[0]
    for j = 1 to 1000 {
        if j == 1 {
            CT[1] = AES-CBC-Encrypt(Key[i], IV[i], PT[1])
            PT[2] = IV[i]
        } else {
            CT[j] = AES-CBC-Encrypt(Key[i], PT[j])
            PT[j+1] = CT[j-1]
        }
    }
    Output CT[1000]

    If KeySize == 128 { Key[i+1] = Key[i] xor CT[1000] }
    If KeySize == 256 { Key[i+1] = Key[i] xor ((CT[999] << 128) | CT[1000]) }

    IV[i+1] = CT[1000]
    PT[0] = CT[999]
}
```

The ciphertext computed in the 1000th iteration (CT[1000]) is the result for each of the 100 3-tuples for each selected key size. 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 above, exchanging CT and PT, and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

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.*

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

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

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

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

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

Three data unit (i.e., plaintext) lengths. *One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 216 bits, whichever is smaller.*

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

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

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

Summary

An ACVP test harness extension was developed by the lab to perform ACVP-like tests for AES-XTS with internally generated tweak values. This tool compares the output generated by the TOE with the known good output of OpenSSL 3. The evaluator used this tool to verify the output generated by the TOE is identical to the output of OpenSSL 3, which confirms the correct implementation of AES-XTS in the TOE.

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

The following tests are conditional based upon the selections made in the SFR.

AES-CBC Tests

For the AES-CBC tests described below, the plaintext, ciphertext, and IV values shall consist of 128-bit blocks. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known-good implementation.

These tests are intended to be equivalent to those described in NIST's AES Algorithm Validation Suite (AESAVS) (http://csrc.nist.gov/groups/STM/cavp/documents/aes/AESAVS.pdf). Known answer values tailored to exercise the AES-CBC implementation can be obtained using NIST's CAVS Algorithm Validation Tool or from NIST's ACPV service for automated algorithm tests (acvp.nist.gov), when available. It is not recommended that evaluators use values obtained from static sources such as the example NIST's AES Known Answer Test Values from the AESAVS document, or use values not generated expressly to exercise the AES-CBC implementation.

AES-CBC Known Answer Tests

KAT-1 (GFSBox):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five different plaintext values for each selected key size 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.

To test the decrypt functionality of AES-CBC, the evaluator shall supply a set of five different ciphertext values for each selected key size and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using a key value of all zeros and an IV of all zeros.

KAT-2 (KeySBox):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five different key values for each selected key size 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.

To test the decrypt functionality of AES-CBC, the evaluator shall supply a set of five different key values for each selected key size and obtain the plaintext that results from AES-CBC decryption of an all-zeros ciphertext using the given key and an IV of all zeros.

KAT-3 (Variable Key):

To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of keys for each selected key size (as described below) and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using each key and an IV of all zeros.

Key i in each set shall have the leftmost i bits set to ones and the remaining bits to zeros, for values of i from 1 to the key size. The keys and corresponding ciphertext are listed in AESAVS, Appendix E.

To test the decrypt functionality of AES-CBC, the evaluator shall use the same keys as above to decrypt the ciphertext results from above. Each decryption should result in an all-zeros plaintext.

KAT-4 (Variable Text):

To test the encrypt functionality of AES-CBC, for each selected key size, the evaluator shall supply a set of 128-bit plaintext values (as described below) and obtain the ciphertext values that result from AES-CBC encryption of each plaintext value using a key of each size and IV consisting of all zeros.

Plaintext value i shall have the leftmost i bits set to ones and the remaining bits set to zeros, for values of i from 1 to 128. The plaintext values are listed in AESAVS, Appendix D.

To test the decrypt functionality of AES-CBC, for each selected key size, use the plaintext values from above as ciphertext input, and AES-CBC decrypt each ciphertext value using key of each size consisting of all zeros and an IV of all zeros.

AES-CBC Multi-Block Message Test The evaluator shall test the encrypt functionality by encrypting nine i-block messages for each selected key size, for 2 <= i <= 10. For each test, the evaluator shall supply a key, an IV, and a plaintext message of length i blocks, and encrypt the message using AES- CBC. The resulting ciphertext values shall be compared to the results of encrypting the plaintext messages using a known good implementation.

The evaluator shall test the decrypt functionality by decrypting nine i-block messages for each selected key size, for 2 <= i <= 10. For each test, the evaluator shall supply a key, an IV, and a ciphertext message of length i blocks, and decrypt the message using AES- CBC. The resulting plaintext values shall be compared to the results of decrypting the ciphertext messages using a known good implementation.

AES-CBC Monte Carlo Tests 236 The evaluator shall test the encrypt functionality for each selected key size using 100 3-tuples of pseudo-random values for plaintext, IVs, and keys.

The evaluator shall supply a single 3-tuple of pseudo-random values for each selected key size. This 3-tuple of plaintext, IV, and key is provided as input to the below algorithm to generate the remaining 99 3-tuples, and to run each 3-tuple through 1000 iterations of AES-CBC encryption.

# Input: PT, IV, Key Key[0] = Key IV[0] = IV PT[0] = PT for i = 1 to 100 { Output Key[i], IV[i], PT[0] for j = 1 to 1000 { if j == 1 { CT[1] = AES-CBC-Encrypt(Key[i], IV[i], PT[1]) PT[2] = IV[i] } else { CT[j] = AES-CBC-Encrypt(Key[i], PT[j]) PT[j+1] = CT[j-1] } } Output CT[1000] If KeySize == 128 { Key[i+1] = Key[i] xor CT[1000] } If KeySize == 256 { Key[i+1] = Key[i] xor ((CT[999] << 128) | CT[1000]) } IV[i+1] = CT[1000] PT[0] = CT[999] }

The ciphertext computed in the 1000th iteration (CT[1000]) is the result for each of the 100 3-tuples for each selected key size. 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 above, exchanging CT and PT, and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

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.

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

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

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

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

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

Three data unit (i.e., plaintext) lengths. One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 216 bits, whichever is smaller.

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

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

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

Summary

An ACVP test harness extension was developed by the lab to perform ACVP-like tests for AES-XTS with internally generated tweak values. This tool compares the output generated by the TOE with the known good output of OpenSSL 3. The evaluator used this tool to verify the output generated by the TOE is identical to the output of OpenSSL 3, which confirms the correct implementation of AES-XTS in the TOE.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-F-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_COP.1-F-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.16 Cryptographic Operation (Key Encryption) (FCS_COP.1(g))

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-G-ASE-01

> *The evaluator shall verify the TSS includes a description of the key size used for encryption and the mode used for the key encryption.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_COP.1(g). The TOE uses AES-CBC **only** on the Apple Silicon platform. Apple Silicon platform use AES-CBC-256 algorithm according to ISO / IEC 18033-3, ISO/IEC 10116 standards for the Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2] module. On Apple silicon devices, the TOE uses the SEP's hardware AES-CBC-256 key encryption implementation when generating the Unlock Key as described in the FCS_PCC_EXT.1 . The key size supported is 256 bits.

Assurance Activity AA-FDEEECPP-FCS_COP.1-G-ASE-01

> *The evaluator shall verify the TSS includes a description of the key size used for encryption and the mode used for the key encryption.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_COP.1(g). The TOE uses AES-CBC **only** on the Apple Silicon platform. Apple Silicon platform use AES-CBC-256 algorithm according to ISO / IEC 18033-3, ISO/IEC 10116 standards for the Apple corecrypto Module 13.0 [Apple silicon, Secure Key Store, Hardware, SL2] module. On Apple silicon devices, the TOE uses the SEP's hardware AES-CBC-256 key encryption implementation when generating the Unlock Key as described in the FCS_PCC_EXT.1 . The key size supported is 256 bits.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-G-AGD-01

> *If multiple key encryption modes are supported, the evaluator examines the guidance documentation to determine that the method of choosing a specific mode/key size by the end user is described.*

Summary

Section 1.1 of [CCGuide], *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Assurance Activity AA-FDEEECPP-FCS_COP.1-G-AGD-01

> *If multiple key encryption modes are supported, the evaluator examines the guidance documentation to determine that the method of choosing a specific mode/key size by the end user is described.*

Summary

This SFR is selection-based in [FDE_EE] and not claimed in [ST] from this PP, but from [FDE_AA] and thus addressed in AA-FDEAACPP-FCS_COP.1-G-AGD-01. This work unit from [FDE_EE] is therefore not applicable and considered implicitly satisfied.

Test Assurance Activities

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

> *The AES test should be followed in FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption.*

Summary

This test is covered by CAVP certs A3496, A1469, A4254, A4104, and C330.

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

> *The AES test should be followed in FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption.*

Summary

This test is covered by CAVP certs A3496, A1469, A4254, A4104, and C330.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_COP.1-G-AKM-01

> *The evaluator shall examine the vendor's KMD to verify that it includes a description of how key encryption will be used as part of the key chain.*

Summary

Section 2.1.1 *Key Chain from Password to BEV* of [KMD] describes the process of producing BEV from the user's password.

1. The TOE performs one round of PBKDF2 operation on the user's password to obtain a 256-bit key.
2. This 256 bit key is repetitively encrypted with AES-CBC-256 cipher using the UID as the key. This iterative encryption process lasts between 100 and 150 ms.
3. The final 256-bit result is termed the REK, which is designated as the BEV and submask.

The TSS for FCS_COP.1(g) in Section 7 *TOE Summary Specification* of [ST] specifies that the TOE performs AES-CBC-256 encryption meeting the standards of ISO/IEC 18033-3 (AES) and ISO/IEC 10116 (CBC).

Assurance Activity AA-FDEEECPP-FCS_COP.1-G-AKM-01

> *The evaluator shall examine the vendor's KMD to verify that it includes a description of how key encryption will be used as part of the key chain.*

Summary

FCS_COP.1(g) is a selection-based SFR in the [FDE_EE]☝ cPP and the [ST]☝ does not select this SFR for the TOE's EE component. This work unit is not applicable and therefore considered to be satisfied.

2.2.1.17 Cryptographic Key Derivation (FCS_KDF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KDF_EXT.1-ASE-01

> *The evaluator shall verify 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 and SP 800-132.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_KDF_EXT.1. The TOE support key derivation functions according to the SP800-132. For password-based authentication, the Secure Enclave implements PBKDF2 to derive the BEV key from the user's password. The PBKDF2 is implemented as specified in SP800-132 following "Option 2b" defined in section 5.4 of the standard. It uses HMAC-SHA-256 as the pseudorandom function (PRF).

Assurance Activity AA-FDEEECPP-FCS_KDF_EXT.1-ASE-01

> *The evaluator shall verify 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 and SP 800-132.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_KDF_EXT.1. The TOE support key derivation functions according to the SP800-132. The Secure Enclave implements PBKDF2 to derive the BEV key from a the user's password. The PBKDF2 is implemented as specified in SP800-132 following "Option 2b" defined in section 5.4 of the standard. It uses HMAC-SHA-256 as the pseudorandom function (PRF).

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KDF_EXT.1-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_KDF_EXT.1-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KDF_EXT.1-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_KDF_EXT.1-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KDF_EXT.1-AKM-01

> *The evaluator shall examine the vendor's KMD to ensure that all keys used are derived using an approved method and a description of how and when the keys are derived.*

Summary

Section 2.1.1 *Key Chain from Password to BEV* of [KMD] describes the process of producing BEV from the user's password.

1. The TOE performs one round of PBKDF2 operation on the user's password to obtain a 256-bit key.
2. This 256 bit key is repetitively encrypted with AES-CBC-256 cipher using the UID as the key. This iterative encryption process lasts between 100 and 150 ms.
3. The final 256-bit result is termed the REK, which is designated as the BEV and submask.

In the TOE, the PBKDF2 is implemented as specified in NIST SP 800-132 following "Option 2b" defined in section 5.4 of the standard. It uses HMAC-SHA-256 as the pseudorandom function (PRF).

Assurance Activity AA-FDEEECPP-FCS_KDF_EXT.1-AKM-01

> *The evaluator shall examine the vendor's KMD to ensure that all keys used are derived using an approved method and a description of how and when the keys are derived.*

Summary

FCS_KDF_EXT.1 is a selection-based SFR in the [FDE_EE] cPP and the [ST] does not select this SFR for the TOE's EE component. This work unit is not applicable and therefore considered to be satisfied.

2.2.1.18 Key Chaining (Initiator) (FCS_KYC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KYC_EXT.1-ASE-01

> *The evaluator shall verify the TSS contains a high-level description of the BEV sizes — that it supports BEV outputs of no fewer 128 bits for products that support only AES-128, and no fewer than 256 bits for products that support AES-256.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_KYC_EXT.1. The TOE supports BEV sizes of 256 bits. As a key chaining initiator, the TOE maintains a key chain of one, using a submask as the BEV. As a key chaining recipient, the TOE maintains a chain of intermediary keys originating from the BEV to the DEK using the following methods:

- symmetric key generation as specified in FCS_CKM.1(b)
- key wrapping as specified in FCS_COP.1(d).

The chain of intermediary keys maintains an effective strength of 256 bits for symmetric keys.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KYC_EXT.1-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KYC_EXT.1-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_KYC_EXT.1-AKM-01

> *The evaluator shall examine the KMD describes a high level description of the key hierarchy for all authorizations methods selected in FCS_AFA_EXT.1 that are used to protect the BEV. The evaluator shall examine the KMD to ensure it describes the key chain in detail. The description of the key chain shall be reviewed to ensure it maintains a chain of keys using key wrap or key derivation methods that meet FCS_COP.1(d) and FCS_KDF_EXT.1.*
>
> *The evaluator shall examine the KMD to ensure that it describes how the key chain process functions, such that it does not expose any material that might compromise any key in the chain. (e.g. using a key directly as a compare value against a TPM) This description must include a diagram illustrating the key hierarchy implemented and detail where all keys and keying material is stored or what it is derived from. The evaluator shall examine the key hierarchy to ensure that at no point the chain could be broken without a cryptographic exhaust or the initial authorization value and the effective strength of the BEV is maintained throughout the key chain.*
>
> *The evaluator shall verify the KMD includes a description of the strength of keys throughout the key chain.*

Summary

Section 2.1 *Key Hierarchy* of [KMD]☝ provides a detailed description of the key hierarchy from the user's password to BEV then to DEK. In particular, Section 2.1.1 *Key Chain from Password to BEV* describes the key chain from the user's password to BEV, where PBKDF2 algorithm is used as the key derivation method to process the user's password. The PBKDF2 is based on HMAC-SHA-256 algorithm and meets the standard of NIST SP 800-132. The information of the PBKDF2 is consistent with FCS_KDF_EXT.1 in Section 6.1 *TOE Security Functional Requirements* of [ST]☝.

Furthermore, Section 2.1.1 of [KMD]☝ provides a diagram illustrating the key chain. The diagram shows the following keys and key materials contributing to the BEV:

- UID,
- Salt,
- Password.

The diagram also indicates the strength of the keys in the key chain:

- UID: 256 bits;
- Salt: 128 bits;
- REK (designated as BEV and submask): 256 bits.

Section 2.2 *Cryptographic Keys* of [KMD]☝ describes the cryptographic keys used in the TOE. In particular, Subsection 2.2.4 describes how each key is generated, and where that key is stored in volatile and non-volatile memories. Please refer to Table 5 and Table 6 for details.

2.2.1.19 Key Chaining (Recipient) (FCS_KYC_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_KYC_EXT.2-ASE-01

> *There are no TSS evaluation activities for this SFR.*

Summary

This work unit has been covered in conjuntion to AA-FDEAACPP-FCS_KYC_EXT.1-ASE-01.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_KYC_EXT.2-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_KYC_EXT.2-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FCS_KYC_EXT.2-AKM-01

> *The evaluator shall examine the KMD to ensure it describes a high level key hierarchy and details of the key chain. The description of the key chain shall be reviewed to ensure it maintains a chain of keys using key wrap or key derivation methods that meet FCS_KDF_EXT.1, FCS_COP.1(d), FCS_COP.1(e), and/or FCS_COP.1(g).*
>
> *The evaluator shall examine the KMD to ensure that it describes how the key chain process functions, such that it does not expose any material that might compromise any key in the chain. (e.g. using a key directly as a compare value against a TPM) This description must include a diagram illustrating the key hierarchy implemented and*

> *detail where all keys and keying material is stored or what it is derived from. The evaluator shall examine the key hierarchy to ensure that at no point the chain could be broken without a cryptographic exhaust or knowledge of the BEV and the effective strength of the DEK is maintained throughout the Key Chain.*
>
> *The evaluator shall verify the KMD includes a description of the strength of keys throughout the key chain.*

Summary

Section 2.1 *Key Hierarchy* of [KMD]☝ provides a detailed description of the key hierarchy from the user's password to BEV then to DEK. In particular, Section 2.1.2 *Key Chain from BEV to DEK* describes the key chain from the BEV to DEK, where the AES-KW algorithm is used as the key wrapping method to unwrap the DEK. The AES-KW algorithm is based on AES-256 algorithm and meets the standard of NIST SP 800-38F. The information of AES-KW algorithm is consistent with FCS_COP.1(d) in Section 6.1 *TOE Security Functional Requirements* of [ST]☝.

The key chain from the BEV to DEK is different on Apple silicon Macs and "Intel with T2" Macs. Section 2.1.2 of [KMD]☝ provides two diagrams to illustrate the key chains on those two types of Macs, respectively. The diagrams show all the keys and key materials contributing to the unwrapping of the DEK:

- Apple silicon Macs:
    - BEV,
    - Media key,
    - Class C key (KEK).
- "Intel with T2" Macs:
    - BEV,
    - Media key.

The diagrams also indicate the strength of the keys in the key chains:

- Apple silicon Macs:
    - BEV: 256 bits;
    - Media key: 256 bits;
    - Class C key (KEK): 256 bits;
    - Per-file key (DEK): 256 bits.
- "Intel with T2" Macs:
    - BEV: 256 bits;
    - Media key: 256 bits;
    - Volume key (DEK): 256 bits.

Section 2.2 *Cryptographic Keys* of [KMD]☝ describes the cryptographic keys used in the TOE. In particular, Subsection 2.2.4 describes how each key is generated, and where that key is stored in volatile and non-volatile memories. Please refer to Table 5 and Table 6 for details.

2.2.1.20 Cryptographic Password Construct and Conditioning (FCS_PCC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_PCC_EXT.1-ASE-01

> *The evaluator shall ensure the TSS describes the manner in which the TOE enforces the construction of passwords, including the length, and requirements on characters (number and type). The evaluator also verifies that the TSS provides a description of how the password is conditioned and the evaluator ensures it satisfies the requirement.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_PCC_EXT.1. On both Apple silicon and Intel with T2 devices, the TOE supports password authentication factor. Passwords of up to 256 characters are supported and can be comprised of any combination of uppercase characters, lowercase characters, numbers, and any other 8-bit special character. This section further describes how the password is conditioned which the evaluator found to adequately satisfies the requirement.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_PCC_EXT.1-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_PCC_EXT.1-ATE-01

> *The evaluator shall also perform the following tests:*
>
> - *Test 1: Ensure that the TOE supports passwords/passphrases of a minimum length of 64 characters.*
> - *Test 2: If the TOE supports a password/passphrase length up to a maximum number of characters, n (which would be greater than 64), then ensure that the TOE will not accept more than n characters.*
> - *Test 3: Ensure that the TOE supports passwords consisting of all characters assigned and supported by the ST author.*

Summary

Test 1: the evaluator changed the password to a password of length 64 characters. The evaluator verified that authentication with the 64-character password is successful

Test 2: the TSS specifies that the TOE supports passwords up to 256 characters. The evaluator attempted to change the password to a password of length 256 characters, and observed the password change failed.

Test 3: the TSS specifies that the TOE supports passwords consisting of all 8-bit values. The evaluator changed the password to a password consisting of uppercase and lowercase alphabetic, numeric, and various special characters. The evaluator verified that authentication with new password is successful.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_PCC_EXT.1-AKM-01

> *The evaluator shall examine the KMD to ensure that the formation of the BEV and intermediary keys is described and that the key sizes match that selected by the ST author.*
>
> *The evaluator shall check that the KMD describes the method by which the password/passphrase 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 KMD 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 BEV as specified above.*

Summary

Section 2.1.1 *Key Chain from Password to BEV* of [KMD]☝ describes the process of producing BEV from the user's password.

1. The TOE performs one round of PBKDF2 operation on the user's password to obtain a 256-bit key.
2. This 256 bit key is repetitively encrypted with AES-CBC-256 cipher using the UID as the key. This iterative encryption process lasts between 100 and 150 ms.
3. The final 256-bit result is termed the REK, which is designated as the BEV and submask. So the size of BEV is 256 bits, matching with the information provided in Section 7 *TOE Summary Specification* of [ST]☝.

Section 2.1.1 of [KMD]☝ also elaborates the implementation of PBKDF2 algorithm in the TOE. The PBKDF2 is implemented as specified in NIST SP 800-132 following "Option 2b" defined in section 5.4 of the standard. It uses HMAC-SHA-256 as the pseudorandom function (PRF). The inputs to the PBKDF2 are:

- The 128-bit salt generated by the TRNG;
- The user's password without any pre-processing;
- The iteration count of one.

The output is the 256-bit key mentioned in Step 1 above.

2.2.1.21 Random Bit Generation (FCS_RBG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_RBG_EXT.1-ASE-01

> *For any RBG services provided by a third party, the evaluator shall ensure the TSS includes a statement about the expected amount of entropy received from such a source, and a full description of the processing of the output of the third-party source. The evaluator shall verify that this statement is consistent with the selection made in FCS_RBG_EXT.1.2 for the seeding of the DRBG. If the ST specifies more than one DRBG, the evaluator shall examine the TSS to verify that it identifies the usage of each DRBG mechanism.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_RBG.EXT.1. On both Apple silicon and Intel T2 devices, the TOE performs deterministic random bit generation services according to NIST SP 800-90A using CTR_DRBG(AES). The DRBG is implemented in the hardware and is part of the SEP's TRNG, which is an entropy source based on hardware noise source.

The SEP's TRNG consists of a hardware noise source produced by 24 ring oscillators, which produces noise that is collected by a SHA-256 conditioner, which is vetted conditioning component per NIST SP800-90B.

Assurance Activity AA-FDEEECPP-FCS_RBG_EXT.1-ASE-01

> *For any RBG services provided by a third party, the evaluator shall ensure the TSS includes a statement about the expected amount of entropy received from such a source, and a full description of the processing of the output of the third-party source. The evaluator shall verify that this statement is consistent with the selection made in FCS_RBG_EXT.1.2 for the seeding of the DRBG. If the ST specifies more than one DRBG, the evaluator shall examine the TSS to verify that it identifies the usage of each DRBG mechanism.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_RBG.EXT.1. On both Apple silicon and Intel T2 devices, the TOE performs deterministic random bit generation services according to NIST SP 800-90A using CTR_DRBG(AES). The DRBG is implemented in the hardware and is part of the SEP's TRNG, which is an entropy source based on hardware noise source.

The SEP's TRNG consists of a hardware noise source produced by 24 ring oscillators, which produces noise that is collected by a SHA-256 conditioner, which is vetted conditioning component per NIST SP800-90B.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_RBG_EXT.1-AGD-01

> *The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected DRBG mechanism(s), if necessary, and provides information regarding how to instantiate/call the DRBG for RBG services needed in this cPP.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of the DRBG mechanism is necessary or available.

Assurance Activity AA-FDEEECPP-FCS_RBG_EXT.1-AGD-01

> *The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected DRBG mechanism(s), if necessary, and provides information regarding how to instantiate/call the DRBG for RBG services needed in this cPP.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of the DRBG mechanism is necessary or available.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_RBG_EXT.1-ATE-01

> *The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable by the TOE, the evaluator shall perform 15 trials for each configuration. The evaluator shall verify that the instructions in the operational guidance for configuration of the RNG are valid.*
>
> *If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “Generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).*
>
> *If the RNG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the*

instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

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

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

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

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

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

Summary

This test is covered by CAVP certs A3490, A1362, DRBG 2029.

Assurance Activity AA-FDEEECPP-FCS_RBG_EXT.1-ATE-01

The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable by the TOE, the evaluator shall perform 15 trials for each configuration. The evaluator shall verify that the instructions in the operational guidance for configuration of the RNG are valid.

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

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

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

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

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

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

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

Summary

This test is covered by CAVP certs A3490, A1362, DRBG 2029.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_RBG_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_RBG_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.22 Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) (FCS_SNI_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_SNI_EXT.1-ASE-01

> *If salts are used, the evaluator shall ensure the TSS describes how salts are generated. The evaluator shall confirm that the salt is generating using an RBG described in FCS_RBG_EXT.1 or by the Operational Environment. If external function is used for this purpose, the TSS should include the specific API that is called with inputs.*
>
> *If IVs or nonces are used, the evaluator shall ensure the TSS describes how nonces are created uniquely and how IVs and tweaks are handled (based on the AES mode). The evaluator shall confirm that the nonces are unique and the IVs and tweaks meet the stated requirements.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_SNI_EXT.1. The TOE can generate salts, nonces, and tweaks using the SEP's DRBG. The DRBG is seeded by the SEP's hardware TRNG. Salts are 16 bytes and are used with the PBKDF2. Nonces are 8 bytes and are used with the trusted update process. The AES-CBC initialization vector (IV), used when generating the Unlock Key, is non-repeating and unpredictable. Tweaks are used with the AES-XTS mode of operation. The tweak values should be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer. The tweak value is the physical block number of the media on which the file is being written. This ensures that values cannot be negative. The number is incremented based on the block number values. This is supported on both Apple silicon and Intel T2 devices.

Assurance Activity AA-FDEEECPP-FCS_SNI_EXT.1-ASE-01

> *The evaluator shall ensure the TSS describes how salts are generated. The evaluator shall confirm that the salt is generating using an RBG described in FCS_RBG_EXT.1 or by the Operational Environment. If external function is used for this purpose, the TSS should include the specific API that is called with inputs.*
>
> *The evaluator shall ensure the TSS describes how nonces are created uniquely and how IVs and tweaks are handled (based on the AES mode). The evaluator shall confirm that the nonces are unique and the IVs and tweaks meet the stated requirements.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FCS_SNI_EXT.1. The TOE can generate salts, nonces, and tweaks using the SEP's DRBG. The DRBG is seeded by the SEP's hardware TRNG. Salts are 16 bytes and are used with the PBKDF2. Nonces are 8 bytes and are used with the trusted update process. The AES-CBC initialization vector (IV), used when generating the Unlock Key, is non-repeating and unpredictable. Tweaks are used with the AES-XTS mode of operation. The tweak values should be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer. The tweak value is the physical block number of the media on which the file is being written. This ensures that values cannot be negative. The number is incremented based on the block number values. This is supported on both Apple silicon and Intel T2 devices.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_SNI_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_SNI_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_SNI_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_SNI_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_SNI_EXT.1-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FCS_SNI_EXT.1-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

2.2.1.23 Validation (FCS_VAL_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_VAL_EXT.1-ASE-01

> *The evaluator shall examine the TSS to determine which authorization factors support validation.*
>
> *The evaluator shall examine the TSS to review a high-level description if multiple submasks are used within the TOE, how the submasks are validated (e.g., each submask validated before combining, once combined validation takes place).*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_VAL_EXT.1. On both Apple silicon and Intel T2 devices, the TOE will validate a BEV using key wrap as specified in FCS_COP.1(d). The TOE requires the validation of the BEV prior to allowing access to TSF data after exiting a Compliant power saving state. The TOE will power/reset after 10 consecutive failed validation attempts.

Assurance Activity AA-FDEEECPP-FCS_VAL_EXT.1-ASE-01

> *The evaluator shall examine the TSS to determine which authorization factors support validation.*
>
> *The evaluator shall examine the TSS to review a high-level description if multiple submasks are used within the TOE, how the submasks are validated (e.g., each submask validated before combining, once combined validation takes place).*
>
> *The evaluator shall also examine the TSS to determine that a subset or all of the authorization factors identified in the SFR can be used to exit from a Compliant power saving state.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FCS_VAL_EXT.1/EE. On both Apple silicon and Intel T2 devices, the TOE will validate a BEV using key wrap as specified in FCS_COP.1(d). The TOE requires the validation of the BEV prior to allowing access to TSF data after exiting a Compliant power saving state. The TOE will power/reset after 10 consecutive failed validation attempts.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_VAL_EXT.1-AGD-01

> *(conditional) If the validation functionality is configurable, the evaluator shall examine the operational guidance to ensure it describes how to configure the TOE to ensure the limits regarding validation attempts can be established.*
>
> *(conditional) If the validation functionality is specified by the ST author, the evaluator shall examine the operational guidance to ensure that it states the values that the TOE uses for limits regarding validation attempts.*

Summary

Validation attempt limits are not configurable.

Section 4 of [CCGuide]☝, *Usage*, states that when the system boots, the user will be prompted to select their account and enter their password authentication factor to unlock the disk, thus exiting the compliant power saving state. After 10 consecutive failed attempts to unlock the disk, FileVault requires the system to be rebooted into recoveryOS. 10 additional authentication attempts are allowed in recoveryOS. FileVault blocks validation once these validation attempts have been exhausted.

Assurance Activity AA-FDEEECPP-FCS_VAL_EXT.1-AGD-01

> *(conditional) If the validation functionality is configurable, the evaluator shall examine the operational guidance to ensure it describes how to configure the TOE to ensure the limits regarding validation attempts can be established.*
>
> *(conditional) If the validation functionality is specified by the ST author, the evaluator shall examine the operational guidance to ensure that it states the values that the TOE uses for limits regarding validation attempts.*
>
> *The evaluator shall verify that the guidance documentation states which authorization factors are allowed to exit a compliant power saving state.*

Summary

Validation attempt limits are not configurable.

Section 4 of [CCGuide]☝, *Usage*, states that when the system boots, the user will be prompted to select their account and enter their password authentication factor to unlock the disk, thus exiting the compliant power saving state. After 10 consecutive failed attempts to unlock the disk, FileVault requires the system to be rebooted into recoveryOS. 10 additional authentication attempts are allowed in recoveryOS. FileVault blocks validation once these validation attempts have been exhausted.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_VAL_EXT.1-ATE-01

> *The evaluator shall perform the following tests:*
>
> *Test 1: The evaluator shall determine the limit on the average rate of the number of consecutive failed authorization attempts. The evaluator will test the TOE by entering that number of incorrect authorization factors in consecutive attempts to access the protected data. If the limit mechanism includes any “lockout” period, the time period tested should include at least one such period. Then the evaluator will verify that the TOE behaves as described in the TSS.*
>
> *Test 2: For each validated authorization factor, ensure that when the user provides an incorrect authorization factor, the TOE prevents the BEV from being forwarded outside the TOE (e.g., to the EE).*

Summary

Test 1: the evaluator attempted to log in to the TOE using a distinct, incorrect password. After 4 consecutive attempts, a time delay of 1 minute was enforced. After 10 consecutive attempts, login is only possible using recoveryOS. Entering recoveryOS requires a power cycle of the TOE, as described in the TSS.

Test 2: using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after an incorrect password is entered, and then after a correct password is entered. The evaluator verified that the required cryptographic keys are not present when an incorrect password is entered.

Assurance Activity AA-FDEEECPP-FCS_VAL_EXT.1-ATE-01

> *The evaluator shall perform the following tests:*
>
> *Test 1: The evaluator shall determine the limit on the average rate of the number of consecutive failed authorization attempts. The evaluator will test the TOE by entering that number of incorrect authorization factors in consecutive attempts to access the protected data. If the limit mechanism includes any “lockout” period, the time period tested should include at least one such period. Then the evaluator will verify that the TOE behaves as described in the TSS.*

> *Test 2: The evaluator shall force the TOE to enter a Compliant power saving state, attempt to resume it from this state, and verify that only a valid authorization factor as defined by the guidance documentation is sufficient to allow the TOE to exit the Compliant power saving state.*

Summary

Test 1: the evaluator attempted to log in to the TOE using a distinct, incorrect password. After 4 consecutive attempts, a time delay of 1 minute was enforced. After 10 consecutive attempts, login is only possible using recoveryOS. Entering recoveryOS requires a power cycle of the TOE, as described in the TSS.

Test 2: using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after an incorrect password is entered, and then after a correct password is entered. The evaluator verified that the required cryptographic keys are not present when an incorrect password is entered.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FCS_VAL_EXT.1-AKM-01

> *The evaluator shall examine the KMD to verify that it described the method the TOE employs to limit the number of consecutively failed authorization attempts.*
>
> *The evaluator shall examine the vendor's KMD to ensure it describes how validation is performed. The description of the validation process in the KMD provides detailed information how the TOE validates the submasks. The KMD describes how the process works, such that it does not expose any material that might compromise the submask(s).*

Summary

Section 2.3 *BEV Validation* of [KMD]☝ describes how the BEV validation is performed. The TOE uses the BEV as the key to unwrap a given ciphertext using SP 800-38F AES-KW with AES-256. If decryption is successful, the BEV is correct. Otherwise, it is not correct.

The ciphertext to be unwrapped is different on Apple silicon and "Intel with T2" Macs. On Apple silicon Macs, the TOE uses the BEV as the key to unwrap the Class C key in the user's keybag. On "Intel with T2" Macs, the TOE use the BEV to unwrap the encrypted form of the volume key.

Section 2.3 of [KMD]☝ also describes the method to limit the number of consecutively failed authorization attempts. The Secure Key Store (SKS) app is part of the Secure Enclave. SKS maintains a timer value and counter of the failed consecutive logins. SKS will delay the unwrap operation as the number of consecutive failed authentication attempts increases. After 10 consecutive failed login attempts, SKS will never unwrap the keys for BEV validation.

Assurance Activity AA-FDEEECPP-FCS_VAL_EXT.1-AKM-01

> *The evaluator shall examine the KMD to verify that it described the method the TOE employs to limit the number of consecutively failed authorization attempts.*
>
> *The evaluator shall examine the vendor's KMD to ensure it describes how validation is performed. The description of the validation process in the KMD provides detailed information how the TOE validates the BEV.*
>
> *The KMD describes how the process works, such that it does not expose any material that might compromise the submask(s).*

Summary

The TOE consists of both FDE AA and FDE EE components. The BEV validation processes in both AA and EE components are implemented as a single process in the TOE.

Section 2.3 *BEV Validation* of [KMD]☝ describes how the BEV validation is performed. The TOE uses the BEV as the key to unwrap a given ciphertext using SP 800-38F AES-KW with AES-256. If decryption is successful, the BEV is correct. Otherwise, it is not correct.

The ciphertext to be unwrapped is different on Apple silicon and "Intel with T2" Macs. On Apple silicon Macs, the TOE uses the BEV as the key to unwrap the Class C key in the user's keybag. On "Intel with T2" Macs, the TOE use the BEV to unwrap the encrypted form of the volume key.

Section 2.3 of [KMD]☝ also describes the method to limit the number of consecutively failed authorization attempts. The Secure Key Store (SKS) app is part of the Secure Enclave. SKS maintains a timer value and counter of the failed consecutive logins. SKS will delay the unwrap operation as the number of consecutive failed authentication attempts increases. After 10 consecutive failed login attempts, SKS will never unwrap the keys for BEV validation.

2.2.2 User data protection (FDP)

2.2.2.1 Protection of Data on Disk (FDP_DSK_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FDP_DSK_EXT.1-ASE-01

> *The evaluator shall examine the TSS to ensure that the description is comprehensive in how the data is written to the disk and the point at which the encryption function is applied. The TSS must make the case that standard methods of accessing the disk drive via the host platforms operating system will pass through these functions.*
>
> *For the cryptographic functions that are provided by the Operational Environment, the evaluator shall check the TSS to ensure it describes, for each platform identified in the ST, the interface(s) used by the TOE to invoke this functionality.*
>
> *The evaluator shall verify the TSS in performing the evaluation activities for this requirement. The evaluator shall ensure the comprehensiveness of the description, confirms how the TOE writes the data to the disk drive, and the point at which it applies the encryption function.*
>
> *The evaluator shall verify that the TSS describes the initialization of the TOE and the activities the TOE performs to ensure that it encrypts all the storage devices entirely when a user or administrator first provisions the TOE. The evaluator shall verify the TSS describes areas of the disk that it does not encrypt (e.g., portions associated with the Master Boot Records (MBRs), boot loaders, partition tables, etc.). If the TOE supports multiple disk encryptions, the evaluator shall examine the administration guidance to ensure the initialization procedure encrypts all storage devices on the platform.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FDP_DSK_EXT.1, and has found the TSS description comprehensive in how the data is written to the disk and the point at which the encryption function is applied. On both Apple silicon and Intel T2 devices,the TOE provides a dedicated AES-XTS crypto engine built into the Direct Memory Access (DMA) path between the flash storage and the main memory of the host platform. This Storage Controller is placed in the middle of the data path between the application processor and the storage device. The Storage Controller performs the encryption/ decryption of the data prior to reaching the application processor or the storage. When a read operation is made, the data must first be decrypted by the Storage Controller before the application processor has access to the data. When a write operation is made, the data is first encrypted by the Storage Controller and then written to storage as a block of encrypted data. This arrangement ensures that standard methods of accessing the storage drive via the operating system will pass through these functions.

When the host platform is provisioned at first run, the user is prompted to enable the TOE's embedded FDE encryption management program (FileVault) and enter a username and password. Once enabled, the storage drive of the host platform remains encrypted and protected from unauthorized access; even if the physical storage device is removed and connected to another host platform. The entire storage drive is encrypted with the exception of the following: partition table, Extensible Firmware Interface (EFI) service partition, Apple File System (APFS) container metadata (allocation bitmaps, checkpoint area, EFI jumpstart driver storage, container locker area), recovery volumes, pre-boot volumes, virtual machine (VM) volumes (used by macOS for strong encrypted swap files), and CoreDump partitions (if present).

Valid credentials are required to be entered before the drive will be decrypted. If the user does not enable FileVault when provisioning the host platform at first run, FileVault can be enabled later through the System Settings » Privacy & Security menu available via the host platform. By default, the host platform's storage drive is always encrypted. The TOE cryptographic key management changes after enabling FileVault.

A recovery key is a randomly generated 28-character code that the user can use to reset their password. The recovery key is generated during the process and manually saved by the user. The recovery key is never stored in the TOE. The recovery key is hashed (SHA-256) and the resulting value is stored in the Secure Enclave. If FileVault is disabled and re-enabled, a new recovery key is generated.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FDP_DSK_EXT.1-AGD-01

> *The evaluator shall review the AGD guidance to determine that it describes the initial steps needed to enable the FDE function, including any necessary preparatory steps. The guidance shall provide instructions that are sufficient, on all platforms, to ensure that all hard drive devices will be encrypted when encryption is enabled.*

Summary

Section 2 of [CCGuide]☝ states that Apple macOS 13 Ventura: FileVault comes pre-installed on the hardware platforms listed in Table 2 - Hardware Platforms" in [CCGuide]☝. Should the need arise, the administrator can manually download and re-install and update the TOE on the supporting hardware. Once FileVault is installed, an administrator can enable FileVault through the following steps:

1. Open the System Settings app
2. Select Privacy & Security
3. Under FileVault, click Turn On
4. Authenticate as an administrator when prompted
5. Do not select "Allow my iCloud account to unlock my disk." This functionality is unevaluated

User data is always encrypted by the SEP. Enabling FileVault, as described in section 3.1, enables a password authentication factor.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FDP_DSK_EXT.1-ATE-01

> *The evaluator shall perform the following tests:*
>
> *Test 1: Write data to random locations, perform required actions and compare:*
>
> - *Ensure TOE is initialized and, if hardware, encryption engine is ready;*
> - *Provision TOE to encrypt the storage device. For SW Encryption products, or hybrid products use a known key and the developer tools.*
> - *Determine a random character pattern of at least 64 KB;*
> - *Retrieve information on what the device TOE's lowest and highest logical address is for which encryption is enabled.*
>
> *Test 2: Write pattern to storage device in multiple locations:*

- *For HW Encryption, randomly select several logical address locations within the device's lowest to highest address range and write pattern to those addresses;*
- *For SW Encryption, write the pattern using multiple files in multiple logical locations.*

Test 3: Verify data is encrypted:

- *For HW Encryption:*
  - *engage device's functionality for generating a new encryption key, thus performing an erase of the key per FCS_CKM.4(a);*
  - *Read from the same locations at which the data was written;*
  - *Compare the retrieved data to the written data and ensure they do not match*
- *For SW Encryption, using developer tools;*
  - *Review the encrypted storage device for the plaintext pattern at each location where the file was written and confirm plaintext pattern cannot be found.*
  - *Using the known key, verify that each location where the file was written, the plaintext pattern can be correctly decrypted using the key.*
  - *If available in the developer tools, verify there are no plaintext files present in the encrypted range.*

Summary

Test 1: the evaluator succesfully created an encrypted partition. In addition, the evaluator created a test file containing 64 KB of random hexadecimal characters, to be used for test 2 and 3.

Test 2: the evaluator copied the test file to the encrypted partition. Then, the evaluator unmounted the partition and exported it as a raw disk dump. The evaluator verified that the random character pattern was not present in the dump of the partition.

Test 3: the evaluator successfully created a second encrypted partition and copied the test file to this partition. Then, the evaluator unmounted the second partition and exported it as a raw disk dump. The evaluator verified that the random character pattern was not present in the second dump. Moreover, the disk dumps of the two test partitions are different, indicating the data is encrypted.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FDP_DSK_EXT.1-AKM-01

The evaluator shall verify the KMD includes a description of the data encryption engine, its components, and details about its implementation (e.g. for hardware: integrated within the device's main SOC or separate co-processor, for software: initialization of the product, drivers, libraries (if applicable), logical interfaces for encryption/decryption, and areas which are not encrypted (e.g. boot loaders, portions associated with the Master Boot Record (MBRs), partition tables, etc.)). The evaluator shall verify the KMD provides a functional (block) diagram showing the main components (such as memories and processors) and the data path between, for hardware, the device's host interface and the device's persistent media storing the data, or for software, the initial steps needed to the activities the TOE performs to ensure it encrypts the storage device entirely when a user or administrator first provisions the product. The hardware encryption diagram shall show the location of the data encryption engine within the data path. The evaluator shall validate that the hardware encryption diagram contains enough detail showing the main components within the data path and that it clearly identifies the data encryption engine.

The evaluator shall verify the KMD provides sufficient instructions for all platforms to ensure that when the user enables encryption, the product encrypts all hard storage devices. The evaluator shall verify that the KMD describes the data flow from the device's host interface to the device's persistent media storing the data. The evaluator shall verify that the KMD provides information on those conditions in which the data bypasses the data encryption engine (e.g. read-write operations to an unencrypted Master Boot Record area).

The evaluator shall verify that the KMD provides a description of the platform's boot initialization, the encryption initialization process, and at what moment the product enables the encryption. The evaluator shall validate that the product does not allow for the transfer of user data before it fully initializes the encryption. The evaluator shall ensure the software developer provides special tools which allow inspection of the encrypted drive either in-band or out-of-band, and may allow provisioning with a known key.

Summary

Section 2.4 *Data Encryption Engine* of [KMD]☝ describes the data encryption engine employed in the TOE. Section 2.4 of [KMD]☝ provides 2 diagrams depicting the main components and the data path of the data encryption engine on Apple silicon and "Intel with T2" Macs, respectively. On both Apple silicon and "Intel with T2" Macs, the TOE provides a dedicated hardware AES-XTS crypto engine built into the Direct Memory Access (DMA) path between the storage device and main memory.

The concrete implementation of the data encryption engine is as follows.

- On Apple silicon Macs, the AES-XTS crypto engine is Apple DMA Storage Controller 2.0 implementing AES-XTS-256 algorithm.
- On "Intel with T2" Macs, the AES-XTS crypto engine is Apple DMA Storage Controller 1.0 implementing AES-XTS-128 algorithm.

Section 2.4 of [KMD]☝ provides the instructions for the user to enable FDE encryption. FileVault can be enabled through the Privacy & Security preference pane in the System Settings app on macOS.

Section 2.4 of [KMD]☝ also lists the portions of the storage device which are excluded from FDE:

- partition table,
- Extensible Firmware Interface (EFI) service partition,
- Apple File System (APFS) container metadata,
- recovery volumes,
- pre-boot volumes,
- virtual machine (VM) volumes,
- CoreDump partitions (if present).

Section 2.5 *Booting* of [KMD]☝ describes the booting procedure of the platform hosting the TOE. The boot procedures on both platforms, Apple silicon and "Intel with T2" Macs, start with the same first step: loading the Boot ROM. The Boot ROM is the immutable code laid down during chip fabrication. The Boot ROM code contains the Apple Root CA public key, which is used to verify the digital signature of each piece of software the platform loads in the bootchain.

The FDE support is enabled right from the start without requiring any configuration or interaction. During boot, the DEK is available only in wrapped form. A user must perform the authentication successfully to unwrap the DEK.

There are currently no tools available to check the encrypted data in the storage drive. Because DMA storage controller key or the volume key are not accessible to the rich OS, there are no tools available on the rich OS which can bypass the encryption/decryption functionality for data.

2.2.3 Security management (FMT)

2.2.3.1 Management of Functions Behavior (FMT_MOF.1)

TSS Assurance Activities

Assurance Activity AA-FDEACPP-FMT_MOF.1-ASE-01

> *If support for Compliant power saving state(s) are claimed in the ST, the evaluator shall ensure the TSS describes how these are managed and shall ensure that TSS describes how only privileged users (administrators) are allowed to manage the states.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FMT_MOF.1. On both Apple silicon and "Intel with T2" Macs, the TOE restricts the ability to modify the behavior of complaint power saving state to authorized users. For more information about Power Saving please see AA-FDEAACPP-FPT_PWR_EXT.1-ASE-01.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_MOF.1-AGD-01

> *The evaluator to check if guidance documentation describes which authorization factors are required to change Compliant power saving state behavior and properties.*

Summary

Per section 1.4 of [CCGuide], the drive is only assumed secure when in a powered-off state up until it is powered on and receives initial authorization.

Section 3.2 of [CCGuide], *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state.

The low-power sleep state can be enabled and disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1 (thus using the administrator password authorization factor).

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_MOF.1-ATE-01

> *The evaluator shall perform the following tests:*
>
> *Test 1 (conditional): If the product supports changes to compliant power saving states, the evaluator presents a privileged authorization credential to the TSF and validates that changes to Compliant power saving state behavior and properties are allowed.*
>
> *Test 2 (conditional): If the product supports changes to compliant power saving states, the evaluator presents a non-privileged authorization credential to the TSF and validates that changes to Compliant power saving state behavior are not allowed.*

Summary

The TOE does not offer the means to configure the power state, so the factory settings are always enforced. Thus, the test requirement is trivially met.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_MOF.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.3.2 Specification of Management Functions (FMT_SMF.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMF.1-ASE-01

> *If item a) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes how the TOE sends the request to the EE to change the DEK.*

If item b) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes how the TOE sends the request to the EE to cryptographically erase the DEK.

If item c) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes the methods by which users may change the set of all authorization factor values supported.

If item d) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes the process to initiate TOE firmware/software updates.

If item e) is selected in FMT_SMF.1.1: If power saving states can be managed, the evaluator shall ensure that the TSS describes how this is performed, including how the TOE supports disabling certain power saving states if more than one are supported. If additional management functions are claimed in the ST, the evaluator shall ensure the TSS describes the additional functions.

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FMT_SMF.1/EE.

The TOE supports the following management functions:

- Authorization Acquisition:
  - Forwarding requests to change the DEK to the EE:
    The DEK can be changed by starting the Disk Utility and select the appropriate volume to be erased. This forces the TOE to cryptographically erase the DEK and create a new one. Data cannot be recovered after this action.
  - Forwarding requests to cryptographically erase the DEK to the EE:
    The DEK can be cryptographically erased by starting the Disk Utility and select the appropriate volume to be erased.
  - Allowing authorized users to change authorization factors or set of authorization factors used:
    Once the user successfully authenticates to the TOE, the TOE can be configured to change the authorization factors by changing the user password.
  - Configure authorization factors:
    Once the user successfully authenticates to the TOE, the TOE can be configured to change the authorization factors by changing the user password.
- Authorization Acquisition and Encryption Engine:
  - Initiate TOE firmware/software updates:
    The user must successfully login to the TOE before initiating a TOE firmware/software update. After successfully authenticating to the TOE, the user manually downloads the TOE software update(s) Apple support website. Once the update(s) is download, the user needs to initiate the TOE update process.

The TOE cryptographically erases the DEK by destroying the keys used to protect it as described below.

Apple silicon

- A Data Encryption Key (DEK) is generated for each file created in an APFS volume. The DEKs are stored in each file metadata within the volume.
- The DEK is protected by wrapping it with a class C key, which is protected by the BEV key, thus providing data confidentiality based on passcodes. The file metadata where the wrapped DEK is stored is also protected by wrapping it using the Media Key, which provides fast erasure of the data. The Media key is also created when a volume is created or erased, protected by wrapping it with the UID, and stored within the Secure Enclave.

- When deleting or erasing a volume, the Media Key of the volume is securely deleted (i.e. zeroized) by the Secure Enclave. This causes all DEKs to be cryptographically erased; the wrapped DEKs remain within each file metadata but the Media Key used to protect all file metadata is no longer available.

Intel with T2

- A single Data Encryption Key (DEK) is generated for an APFS volume (known as the Volume Key) when a new volume is created or an existing volume is erased. The DEK is stored in the APFS volume.
- The DEK is protected by wrapping the key with the BEV key, which provides data confidentiality based on passcodes. The wrapped key is wrapped again using the Media Key, which provides fast erasure of the data. The Media key is also created when a volume is created or erased, protected by wrapping it with the UID, and stored within the Secure Enclave.
- When deleting or erasing a volume, the Media Key of the volume is securely deleted (i.e. zeroized) by the Secure Enclave. This causes the DEK (i.e. Volume Key) to be cryptographically erased; the DEK value remains in the module but the Media Key used to protect it is no longer available.

Assurance Activity AA-FDEEECPP-FMT_SMF.1-ASE-01

If item a) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes how the TOE changes the DEK.

If item b) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes how the TOE cryptographically erases the DEK.

If item c) is selected in FMT_SMF.1.1: The evaluator shall ensure the TSS describes the process to initiate TOE firmware/software updates.

If item d) is selected in FMT_SMF.1.1: If additional management functions are claimed in the ST, the evaluator shall verify that the TSS describes those functions.

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FMT_SMF.1/EE.

The TOE supports the following management functions:

- Authorization Acquisition:
    - Forwarding requests to change the DEK to the EE:
      The DEK can be changed by starting the Disk Utility and select the appropriate volume to be erased. This forces the TOE to cryptographically erase the DEK and create a new one. Data cannot be recovered after this action.
    - Forwarding requests to cryptographically erase the DEK to the EE:
      The DEK can be cryptographically erased by starting the Disk Utility and select the appropriate volume to be erased.
    - Allowing authorized users to change authorization factors or set of authorization factors used:
      Once the user successfully authenticates to the TOE, the TOE can be configured to change the authorization factors by changing the user password.
    - Configure authorization factors:
      Once the user successfully authenticates to the TOE, the TOE can be configured to change the authorization factors by changing the user password.

- Authorization Acquisition and Encryption Engine:
    - Initiate TOE firmware/software updates:
      The user must successfully login to the TOE before initiating a TOE firmware/ software update. After successfully authenticating to the TOE, the user manually downloads the TOE software update(s) Apple support website. Once the update(s) is download, the user needs to initiate the TOE update process.

The TOE cryptographically erases the DEK by destroying the keys used to protect it as described below.

Apple silicon

- A Data Encryption Key (DEK) is generated for each file created in an APFS volume. The DEKs are stored in each file metadata within the volume.
- The DEK is protected by wrapping it with a class C key, which is protected by the BEV key, thus providing data confidentiality based on passcodes. The file metadata where the wrapped DEK is stored is also protected by wrapping it using the Media Key, which provides fast erasure of the data. The Media key is also created when a volume is created or erased, protected by wrapping it with the UID, and stored within the Secure Enclave.
- When deleting or erasing a volume, the Media Key of the volume is securely deleted (i.e. zeroized) by the Secure Enclave. This causes all DEKs to be cryptographically erased; the wrapped DEKs remain within each file metadata but the Media Key used to protect all file metadata is no longer available.

Intel with T2

- A single Data Encryption Key (DEK) is generated for an APFS volume (known as the Volume Key) when a new volume is created or an existing volume is erased. The DEK is stored in the APFS volume.
- The DEK is protected by wrapping the key with the BEV key, which provides data confidentiality based on passcodes. The wrapped key is wrapped again using the Media Key, which provides fast erasure of the data. The Media key is also created when a volume is created or erased, protected by wrapping it with the UID, and stored within the Secure Enclave.
- When deleting or erasing a volume, the Media Key of the volume is securely deleted (i.e. zeroized) by the Secure Enclave. This causes the DEK (i.e. Volume Key) to be cryptographically erased; the DEK value remains in the module but the Media Key used to protect it is no longer available.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMF.1-AGD-01

If item a) and/or b) is selected in FMT_SMF.1.1: The evaluator shall examine the operational guidance to ensure that it describes how the functions for A and B can be initiated by the user.

If item c) is selected in FMT_SMF.1.1: The evaluator shall examine the operational guidance to ensure that it describes how selected authorization factor values are changed.

If item d) is selected in FMT_SMF.1.1: The evaluator shall examine the operational guidance to ensure that it describes how to initiate TOE firmware/software updates.

If item e) is selected in FMT_SMF.1.1: Default Authorization Factors: It may be the case that the TOE arrives with default authorization factors in place. If it does, then the selection in section E must be made so that there is a mechanism to change these authorization factors. The operational guidance shall describe the method by which the user changes these factors when they are taking ownership of the device. The TSS shall describe the default authorization factors that exist.

Disable Key Recovery: The guidance for disabling this capability shall be described in the AGD documentation.

Power Saving: The guidance shall describe the power saving states that are supported by the TSF, how these states are applied, how to configure when these states are applied (if applicable), and how to enable/disable the use of specific power saving states (if applicable).

Summary

Section 3.3.4 of [CCGuide], *Changing/Erasing the Data Encryption Key*, describes how the DEK can be changed or erased.

Per section 3.3.4, the DEK can be changed/erased by erasing the entire disk. An administrator can erase the disk through the following steps:

1. Follow the steps in [CCGuide] Section 2.3 to boot into recoveryOS
2. Select Options
3. Authenticate as an administrator
4. Launch Disk Utility
5. Select Macintosh HD
6. Click Erase

Section 3.3.3 of [CCGuide] describes how a user can change the password through the following steps:

1. Open the System Settings app
2. Select Users & Groups
3. Click "Change Password"
4. Enter the old password and the new password
5. Click Change Password

Per section 2.2 of [CCGuide], *Update*, FileVault updates are bundled with macOS updates. Any user can check for updates to FileVault by navigating to Apple menu > About This Mac > Software Update or by opening the System Settings app and selecting Software Update. If updates are available, the user will be given the option of installing updates. Users can also check for updates from the command line by issuing the softwareupdate -l command.

The user must be an authorized user (i.e., successfully logged in) to initiate an update.

Key recovery is not a function provided by the TOE. [CCGuide] section 3.3.4, "Changing/Erasing the Data Entryption Key" states the encryption key can only be changed or erased by erasing the entire disk.

Section 3.2 of [CCGuide], *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state.

The low-power sleep state can be disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1. The low-power sleep state can be re-enabled by having an administrator open Terminal and run sudo pmset -a disablesleep 0.

Assurance Activity AA-FDEEECPP-FMT_SMF.1-AGD-01

If item a) is selected in FMT_SMF.1.1: The evaluator shall review the AGD guidance and shall determine that the instructions for changing a DEK exist. The instructions must cover all environments on which the TOE is claiming conformance, and include any preconditions that must exist in order to successfully generate or re-generate the DEK.

If item c) is selected in FMT_SMF.1.1: he evaluator shall examine the operational guidance to ensure that it describes how to initiate TOE firmware/software updates.

If item d) is selected in FMT_SMF.1.1: TDefault Authorization Factors: It may be the case that the TOE arrives with default authorization factors in place. If it does, then the selection in item D must be made so that there is a mechanism to change these authorization factors. The operational guidance shall describe the method by which the user changes these factors when they are taking ownership of the device. The TSS shall describe the default authorization factors that exist.

> *Disable Key Recovery: The guidance for disabling this capability shall be described in the AGD documentation.*

Summary

Section 3.3.4 of [CCGuide]☝, *Changing/Erasing the Data Encryption Key*, describes how the DEK can be changed or erased.

Per section 3.3.4, the DEK can be changed/erased by erasing the entire disk. An administrator can erase the disk through the following steps:

1. Follow the steps in [CCGuide]☝ Section 2.3 to boot into recoveryOS
2. Select Options
3. Authenticate as an administrator
4. Launch Disk Utility
5. Select Macintosh HD
6. Click Erase

Per section 2.2 of [CCGuide]☝, *Update*, FileVault updates are bundled with macOS updates. Any user can check for updates to FileVault by navigating to Apple menu > About This Mac > Software Update or by opening the System Settings app and selecting Software Update. If updates are available, the user will be given the option of installing updates. Users can also check for updates from the command line by issuing the softwareupdate -l command.

The user must be an authorized user (i.e., successfully logged in) to initiate an update.

Key recovery is not a function provided by the TOE. [CCGuide]☝ section 3.3.4, "Changing/Erasing the Data Entryption Key" states the encryption key can only be changed or erased by erasing the entire disk.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMF.1-ATE-01

> *If item a) and/or b) is selected in FMT_SMF.1.1: The evaluator shall verify that the TOE has the functionality to forward a command to the EE to change and cryptographically erase the DEK. The actual testing of the cryptographic erase will take place in the EE.*
>
> *If item c) is selected in FMT_SMF.1.1: The evaluator shall initialize the TOE such that it requires the user to input an authorization factor in order to access encrypted data.*
>
> > *Test 1: The evaluator shall first provision user authorization factors, and then verify all authorization values supported allow the user access to the encrypted data. Then the evaluator shall exercise the management functions to change a user's authorization factor values to a new one. Then he or she will verify that the TOE denies access to the user's encrypted data when he or she uses the old or original authorization factor values to gain access.*
>
> *If item d) is selected in FMT_SMF.1.1: The evaluator shall verify that the TOE has the functionality to initiate TOE firmware/software updates.*
>
> *If item e) is selected in FMT_SMF.1.1: If additional management functions are claimed, the evaluator shall verify that the additional features function as described.*
>
> > *Test 2 (conditional): If the TOE provides default authorization values, the evaluator shall change these values in the course of taking ownership of the device as described in the operational guidance. The evaluator shall then confirm that the (old) authorization values are no longer valid for data access.*
> >
> > *Test 3 (conditional): If the TOE provides key recovery capability whose effects are visible at the TOE interface, then the evaluator shall devise a test that ensures that the key recovery capability has been or can be disabled following the guidance provided by the vendor.*
> >
> > *Test 4 (conditional): If the TOE provides the ability to configure the power saving states that are entered by certain events, the evaluator shall devise a test that causes the TOE to enter a specific power saving state, configure the TSF so that this activity causes a different state to be entered, repeat the activity, and observe the new state is entered as configured.*

> *Test 5 (conditional): If the TOE provides the ability to disable the use of one or more power saving states, the evaluator shall devise a test that enables all supported power saving states and demonstrates that the TOE can enter into each of these states. The evaluator shall then disable the supported power saving states one by one, repeating the same set of actions that were performed at the start of the test, and observe each time that when a power saving state is configured to no longer be used, none of the behavior causes the disabled state to be entered.*

Summary

Using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after an incorrect password is entered, and then after a correct password is entered. The evaluator verified that the required cryptographic keys are not present when an incorrect password is entered.

The evaluator verified that the TOE allows for the update of the macOS version.

The evaluator verified that the TOE allows the password to be changed and authentication with the new password functions correctly.

Test 2 is not applicable because the TOE does not provide default authorization factors.

Test 3 is not applicable because the TOE does not offer key recovery capability.

Test 4 and 5 are not applicable because the TOE does not allow configuring the power saving functionality.

Assurance Activity AA-FDEEECPP-FMT_SMF.1-ATE-01

> *If item a) and/or b) is selected in FMT_SMF.1.1: The evaluator shall verify that the TOE has the functionality to change and cryptographically erase the DEK (effectively removing the ability to retrieve previous user data).*
>
> *If item c) is selected in FMT_SMF.1.1: The evaluator shall verify that the TOE has the functionality to initiate TOE firmware/software updates.*
>
> *If item d) is selected in FMT_SMF.1.1: If additional management functions are claimed, the evaluator shall verify that the additional features function as described.*

Summary

Using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore before and after a factory reset. The evaluator verified that the contents of the key store are different, implying that new cryptographic keys were generated during factory reset and the old cryptographic keys were destroyed.

The evaluator verified that the TOE allows for the update of the macOS version.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMF.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FMT_SMF.1-AKM-01

> *If item d) is selected in FMT_SMF.1.1: If the TOE offers the functionality to import an encrypted DEK, the evaluator shall ensure the KMD describes how the TOE imports a wrapped DEK and performs the decryption of the wrapped DEK.*

Summary

Per FMT_SMF.1/EE in Section 6.1 *TOE Security Functional Requirements* of [ST]☝, the TOE does not offer the functionality to import an encrypted DEK (item d). This work unit is not applicable and therefore considered to be satisfied.

2.2.3.3 Security Roles (FMT_SMR.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMR.1-ASE-01

> *There are no TSS evaluation activities for this SFR. Evaluation of this SFR is performed as part of evaluating FMT_MOF.1 and FMT_SMF.1.*

Summary

This work unit has been covered in conjunction with AA-FDEAACPP-FMT_MOF.1-ASE-01 and AA-FDEAACPP-FMT_SMF.1-ASE-01.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMR.1-AGD-01

> *There are no guidance evaluation activities for this SFR. Evaluation of this SFR is performed as part of evaluating FMT_MOF.1 and FMT_SMF.1.*

Summary

There are no guidance evaluation activities for this SFR. Evaluation of this SFR is performed as part of evaluating FMT_MOF.1 and FMT_SMF.1.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMR.1-ATE-01

> *There are no test evaluation activities for this SFR. Evaluation of this SFR is performed as part of evaluating FMT_MOF.1 and FMT_SMF.1.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FMT_SMR.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.4 Protection of the TSF (FPT)

2.2.4.1 Firmware Update Authentication (FPT_FUA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEEECPP-FPT_FUA_EXT.1-ASE-01

> *The evaluator shall examine the TSS to ensure that it describes how the TOE uses the RTU, what type of key or hash value, and where the value is stored on the RTU. The evaluator shall also verify that the TSS contains a description (storage location) of where the original firmware exists.*

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_FUA_EXT.1. The TOE provides only to the authorized users the ability to query the current version of the TOE software and initiate an updates. The TOE rely on a vendor-controlled (Apple) server to obtaining firmware update code packages. The code packages containing the macOS, T2OS/firmware, and sepOS/firmware are all bundled together. The T2OS and sepOS firmware is stored within the Secure Enclave. The TOE stores the incoming update in a temporary location on flash. Once the transfer is complete, the SEP verifies the digital signature using the algorithm described in FCS_COP.1(a). 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.

Guidance Assurance Activities

Assurance Activity AA-FDEEECPP-FPT_FUA_EXT.1-AGD-01

> *There are no AGD evaluation activities for this SFR.*

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEEECPP-FPT_FUA_EXT.1-ATE-01

> *There are no test evaluation activities for this SFR.*

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEEECPP-FPT_FUA_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.4.2 Protection of Key and Key Material (FPT_KYP_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_KYP_EXT.1-ASE-01

> *[TD0458] The evaluator shall examine the TSS and verify it identifies the methods used to protect keys stored in non-volatile memory.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_KYP_EXT.1/AA. The TOE's Keys for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed.

Assurance Activity AA-FDEEECPP-FPT_KYP_EXT.1-ASE-01

The evaluator shall examine the TSS to verify that it describes the method by which intermediate keys are generated using submask combining.

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_KYP_EXT.1/EE. The TOE's Keys for the UID and the Unlock Key, all symmetric keys are introduced into volatile memory after being randomly generated or by unwrapping or decrypting a key stored in non-volatile memory. The Unlock Key is introduced into volatile memory after the password-based derivation process has been completed.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_KYP_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_KYP_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_KYP_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_KYP_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_KYP_EXT.1-AKM-01

[TD0458] The evaluator shall verify the KMD to ensure it describes the storage location of all keys and the protection of all keys stored in non-volatile memory. The description of the key chain shall be reviewed to ensure the selected method is followed for the storage of wrapped or encrypted keys in non-volatile memory and plaintext keys in non-volatile memory meet one of the criteria for storage.

Summary

Section 2.2 *Cryptographic Keys* of [KMD]☝ describes the cryptographic keys used in the TOE. In particular, Subsection 2.2.4 describes

- where in volatile memory and non-volatile memory the keys are stored, and
- how the keys stored in non-volatile memory are protected.

The keys stored in non-volatile memory are protected by either key wrapping or the Secure Enclave. The following table shows the protection of all keys which are stored in non-volatile memory.

Table 7: Protection of keys stored in non-volatile memory

Key	Storage location	Protection mechanism
Secure Enclave UID	The Secure Enclave ROM	The Secure Enclave. The UID can only be accessed by the Secure Enclave.
Media Key	The Secure Enclave effaceable storage	The Secure Enclave and Cryptographically wrapped. Media Key is wrapped with Secure Enclave UID and can only be accessed by the Secure Enclave.
Class C key	Non-volatile storage of the Operational Environment (TOE device). Stored in the user's keybag which is managed by the rich OS.	Cryptographically wrapped. The Class C key in a user's keybag is wrapped with that user's REK.
Per-file key	Non-volatile storage of the Operational Environment (TOE device). The per-file key for a file is stored in that file's metadata.	Cryptographically wrapped. Per-file key is wrapped with KEK (Class C key) and further wrapped with Media Key.
Volume key	Non-volatile storage of the Operational Environment (TOE device). The volume key for a filesystem volume is stored in that volume.	Cryptographically wrapped. Volume key is wrapped with the REK and further wrapped with Media Key.

Assurance Activity AA-FDEEECPP-FPT_KYP_EXT.1-AKM-01

The evaluator shall examine the KMD for a description of the methods used to protect keys stored in non-volatile memory.

The evaluator shall verify the KMD to ensure it describes the storage location of all keys and the protection of all keys stored in non-volatile memory. The description of the key chain shall be reviewed to ensure the selected method is followed for the storage of wrapped or encrypted keys in non-volatile memory and plaintext keys in non-volatile memory meet one of the criteria for storage.

Summary

The TOE consists of both FDE AA and FDE EE components. The assurance activity for FPT_KYP_EXT.1 is the same on both FDE AA and FDE EE components.

Please refer to Table 7 for the provided information, which covers both FDE AA and FDE EE components.

2.2.4.3 Power Saving States (FPT_PWR_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.1-ASE-01

> *The evaluator shall validate the TSS contains a list of Compliant power saving states.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_PWR_EXT.1/AA. The TOE is compliant to the following power saving state: G2(S5) (soft off).

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.1-ASE-01

> *The evaluator shall validate the TSS contains a list of Compliant power saving states.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_PWR_EXT.1/EE. The TOE is compliant to the following power saving state G2(S5) (soft off).

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.1-AGD-01

> *The evaluator shall ensure that guidance documentation contains a list of Compliant power saving states. If additional power saving states are supported, then the evaluator shall validate that the guidance documentation states how non-Compliant power states are disabled.*

Summary

Section 1.4, "Assumptions and Warnings" in [CCGuide]☝ states the user will not leave the platform in a lock screen or sleep state. Section 3.2, "Sleep State" states that G2(S5) (soft off) is the only Compliant power-saving state supported in the evaluated configuration. It further describes how the administrator must disable sleep state in the evaluated configuration which will only allow the user to leave it in a power off state.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.1-AGD-01

> *The evaluator shall ensure that guidance documentation contains a list of Compliant power saving states. If additional power saving states are supported, then the evaluator shall validate that the guidance documentation states how the use of non-Compliant power saving states can be avoided.*

Summary

Section 1.4, "Assumptions and Warnings" in [CCGuide]☝ states the user will not leave the platform in a lock screen or sleep state. Section 3.2, "Sleep State" states that G2(S5) (soft off) is the only Compliant power-saving state supported in the evaluated configuration. It further describes how the administrator must disable sleep state in the evaluated configuration which will only allow the user to leave it in a power off state.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.1-ATE-01

> *The evaluator shall confirm that for each listed compliant state all key/key materials are removed from volatile memory by using the test defined in FCS_CKM.4(d).*

Summary

The evaluator rebooted the device, causing it to enter and resume from the compliant G2(S5) (power off) power saving state. Then, using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore and verified that the cryptographic keys are not present. This implies that all keys and key material is removed from volatile memory in the compliant power saving state.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.1-ATE-01

> *The evaluator shall confirm that for each listed Compliant state all key/key materials are removed from volatile memory by using the test indicated by the selection in FCS_CKM_EXT.6.*

Summary

The evaluator rebooted the device, causing it to enter and resume from the compliant G2(S5) (power off) power saving state. Then, using specialized debugging tools, the evaluator inspected the contents of the AppleKeystore and verified that the cryptographic keys are not present. This implies that all keys and key material is removed from volatile memory in the compliant power saving state.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.4.4 Timing of Power Saving States (FPT_PWR_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.2-ASE-01

> *The evaluator shall validate that the TSS contains a list of conditions under which the TOE enters a Compliant power saving state.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_PWR_EXT.2. The TOE, on both Apple silicon and Intel T2 devices, supports the following power saving state: G2(S5) (soft off). The user can either select the menu option Apple menu >> Shut Down, or press and hold the physical power button to enter the power saving state.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.2-ASE-01

> *The evaluator shall validate that the TSS contains a list of conditions under which the TOE enters a Compliant power saving state.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_PWR_EXT.2. The TOE, on both Apple silicon and Intel T2 devices, the TOE supports the following power saving state: G2(S5) (soft off). The user can either select the menu option Apple menu >> Shut Down, or press and hold the physical power button to enter the power saving state.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.2-AGD-01

> *The evaluator shall check that the guidance contains a list of conditions under which the TOE enters a Compliant power saving state. Additionally, the evaluator shall verify that the guidance documentation states whether unexpected power-loss events may result in entry to a non-Compliant power saving state and, if that is the case, validate that the documentation contains information on mitigation measures.*

Summary

Per section 1.4 of [CCGuide]☝, the drive is only assumed secure when in a powered-off state up until it is powered on and receives initial authorization.

Section 3.2 of [CCGuide]☝, *Sleep State*, states that macOS has a low-power sleep state that is not used in the evaluated configuration. The drive is only assumed secure when in a powered-off state.

Section 3.2 also states in addition to shutdown, in the event of complete power loss, the TOE enters the Compliant power off state and can only resume by re-authenticating.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.2-AGD-01

> *The evaluator shall check that the guidance contains a list of conditions under which the TOE enters a Compliant power saving state. Additionally, the evaluator shall verify that the guidance documentation provides information on how long it is expected to take for the TOE to fully transition into the Compliant power saving state (e.g. how many seconds for the volatile memory to be completely cleared).*

Summary

Per section 1.4 of [CCGuide]☝, the drive is only assumed secure when in a powered-off state up until it is powered on and receives initial authorization.

Section 3.2 of [CCGuide]☝, *Sleep State*, states that macOS has a low-power sleep state that is not used in the evaluated configuration. The drive is only assumed secure when in a powered-off state.

Section 3.2 also states in addition to shutdown, in the event of complete power loss, the TOE enters the Compliant power off state and can only resume by re-authenticating.

Also provided is a time estimate for the TOE to fully transition into the Compliant power-saving state, thus clearing memory.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.2-ATE-01

> *The evaluator shall trigger each condition in the list of identified conditions and ensure the TOE ends up in a compliant power saving state by running the test identified in FCS_CKM.4(d).*

Summary

The TSS specifies that the TSF enters the compliant power saving state only upon a user-initiated request. While evaluating FPT_PWR_EXT.1, the evaluator performed this request. Therefore, the explanation for FPT_PWR_EXT.1 applies equally to this test.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.2-ATE-01

> *The evaluator shall trigger each condition in the list of identified conditions and ensure the TOE ends up in a Compliant power saving state by using the test indicated by the selection in FCS_CKM_EXT.6.*

Summary

The TSS specifies that the TSF enters the compliant power saving state only upon a user-initiated request. While evaluating FPT_PWR_EXT.1, the evaluator performed this request. Therefore, the explanation for FPT_PWR_EXT.1 applies equally to this test.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_PWR_EXT.2-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_PWR_EXT.2-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.2.4.5 TSF Testing (FPT_TST_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TST_EXT.1-ASE-01

> *The evaluator shall verify that the TSS describes the known-answer self-tests for cryptographic functions.*

> *The evaluator shall verify that the TSS describes, for some set of non-cryptographic functions affecting the correct operation of the TOE and the method by which the TOE tests those functions. The evaluator shall verify that the TSS includes each of these functions, the method by which the TOE verifies the correct operation of the function. The evaluator shall verify that the TSF data are appropriate for TSF Testing. For example, more than blocks are tested for AES in CBC mode, output of AES in GCM mode is tested without truncation, or 512-bit key is used for testing HMAC-SHA-512.*
>
> *If FCS_RBG_EXT.1 is implemented by the TOE and according to NIST SP 800-90, the evaluator shall verify that the TSS describes health tests that are consistent with section 11.3 of NIST SP 800-90.*
>
> *If any FCS_COP functions are implemented by the TOE, the TSS shall describe the known-answer self-tests for those functions.*
>
> *The evaluator shall verify that the TSS describes, for some set of non-cryptographic functions affecting the correct operation of the TSF, the method by which those functions are tested. The TSS will describe, for each of these functions, the method by which correct operation of the function/component is verified. The evaluator shall determine that all of the identified functions/components are adequately tested on start- up.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_TST_EXT.1. The TOE performs the following known answer tests (KATs) to verify the correct operation of the cryptographic functions:

- CTR_DRBG with AES-256: The TOE instantiates the DRBG with a known value, invokes the generate function, and compares the generated bits to the expected bits.
- HMAC-SHA-256: MAC generation with a known key and message.
- AES-CBC 256-bit Encrypt/Decrypt: KATs
- Apple silicon:
  - AES-XTS 256-bit Encrypt/Decrypt: This shows the correct operation of AES Encrypt and Decrypt primitive functions with a 512-bit key.
  - ECDSA P-521 with SHA-512 Signature Verification: Satisfied by the Firmware Integrity signature verification test.
- Intel with T2:
  - AES-XTS 128-bit Encrypt/Decrypt: This shows the correct operation of AES Encrypt and Decrypt primitive functions with a 256-bit key.
  - RSA 4096 with SHA-256 Signature Verification: Satisfied by the Firmware Integrity signature verification test.

In addition, the evaluator has verified that the TOE on both Apple silicon and Intel T2 devices, performs deterministic random bit generation services according to NIST SP 800-90A using CTR_DRBG(AES). The SEP TRNG is seeded by 24 ring oscillators. The ring oscillators are constantly inputting new noise data into the SHA-256 conditioner from which the DRBG seed is obtained. The conditioned output of the TRNG contains 1 bit of entropy per each bit of data. The evaluator taken into account the non-cryptographic functions and how are tested.

For the Apple silicon platform, during power-up, the application processor loads the Boot ROM which contains the Apple Root CA public key. The Boot ROM authenticates the Low-Level Bootloader (LLB) signature using the Apple Root CA public key. The LLB authenticates system-paired firmware signatures. The LLB authenticates iBoot stage 2 signature. Boot stage 2 authenticates the macOS-paired firmware, Boot Kernel Collection, Auxiliary Kernel Collection (if applicable), system trust cache, and signed system volume signatures.

macOS begins execution and authenticates third party kernel extensions (kexts) and OS user space. For the Intel T2 platform, during power-up, the TOE performs a signature verification of firmware and software using the Apple Root CA Public Key. When the Mac is powered-on, the SEP initiates the Secure Boot process. The SEP's Boot ROM first authenticates the signature of the Bridge Boot code (Apple T2 Security Chip Boot ROM code). If the verifications fails, the TOE returns an error and enters the Device Firmware Upgrade (DFU) mode; requiring a correct update to continue.

On both the two platforms, If the verification is successful, the Bridge Boot code then authenticates the signature of the T2 kernel cache. The T2 kernel cache then authenticates the signature of the Unified Extensible Firmware Interface (UEFI) firmware. The UEFI firmware is then used to authenticate the boot.efi file within the Intel processor of the TOE host device. The boot.efi file then authenticates the macOS immutable kernel. The macOS then authenticates third party kernel extensions (kexts) and OS user space.

Assurance Activity AA-FDEEECPP-FPT_TST_EXT.1-ASE-01

> *The evaluator shall verify that the TSS describes the known-answer self-tests for cryptographic functions.*
>
> *The evaluator shall verify that the TSS describes, for some set of non-cryptographic functions affecting the correct operation of the TOE and the method by which the TOE tests those functions. The evaluator shall verify that the TSS includes each of these functions, the method by which the TOE verifies the correct operation of the function. The evaluator shall verify that the TSF data are appropriate for TSF Testing. For example, more than blocks are tested for AES in CBC mode, output of AES in GCM mode is tested without truncation, or 512-bit key is used for testing HMAC-SHA-512.*
>
> *If FCS_RBG_EXT.1 is implemented by the TOE and according to NIST SP 800-90, the evaluator shall verify that the TSS describes health tests that are consistent with section 11.3 of NIST SP 800-90.*
>
> *If any FCS_COP functions are implemented by the TOE, the TSS shall describe the known-answer self-tests for those functions.*
>
> *The evaluator shall verify that the TSS describes, for some set of non-cryptographic functions affecting the correct operation of the TSF, the method by which those functions are tested. The TSS will describe, for each of these functions, the method by which correct operation of the function/component is verified. The evaluator shall determine that all of the identified functions/components are adequately tested on start- up.*

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description. The evaluator examined the provided information for FPT_TST_EXT.1. The TOE performs the following known answer tests (KATs) to verify the correct operation of the cryptographic functions:

- CTR_DRBG with AES-256: The TOE instantiates the DRBG with a known value, invokes the generate function, and compares the generated bits to the expected bits.
- HMAC-SHA-256: MAC generation with a known key and message.
- AES-CBC 256-bit Encrypt/Decrypt: KATs
- Apple silicon:
  - AES-XTS 256-bit Encrypt/Decrypt: this shows the correct operation of AES Encrypt and Decrypt primitive functions with 256-bit keys.
  - ECDSA P-521 with SHA-512 Signature Verification: Satisfied by the Firmware Integrity signature verification test.
- Intel with T2:
  - AES-XTS 128-bit Encrypt/Decrypt: This shows the correct operation of AES Encrypt and Decrypt primitive functions with 128-bit keys.
  - RSA 4096 with SHA-256 Signature Verification: Satisfied by the Firmware Integrity signature verification test.

In addition, the evaluator has verified that the TOE on both Apple silicon and Intel T2 devices, performs deterministic random bit generation services according to NIST SP 800-90A using CTR_DRBG(AES). The SEP TRNG is seeded by 24 ring oscillators. The ring oscillators are constantly inputting new noise data into the SHA-256 conditioner from which the DRBG seed is obtained. The conditioned output of the TRNG contains 1 bit of entropy per each bit of data. The evaluator taken into account the non-cryptographic functions and how are tested.

For the Apple silicon platform, during power-up, the application processor loads the Boot ROM which contains the Apple Root CA public key. The Boot ROM authenticates the Low-Level Bootloader (LLB) signature using the Apple Root CA public key. The LLB authenticates system-paired firmware signatures. The LLB authenticates iBoot stage 2 signature. Boot stage 2 authenticates the macOS-paired firmware, Boot Kernel Collection, Auxiliary Kernel Collection (if applicable), system trust cache, and signed system volume signatures.

macOS begins execution and authenticates third party kernel extensions (kexts) and OS user space. For the Intel T2 platform, during power-up, the TOE performs a signature verification of firmware and software using the Apple Root CA Public Key. When the Mac is powered-on, the SEP initiates the Secure Boot process. The SEP's Boot ROM first authenticates the signature of the Bridge Boot code (Apple T2 Security Chip Boot ROM code). If the verifications fails, the TOE returns an error and enters the Device Firmware Upgrade (DFU) mode; requiring a correct update to continue.

On both the two platforms, If the verification is successful, the Bridge Boot code then authenticates the signature of the T2 kernel cache. The T2 kernel cache then authenticates the signature of the Unified Extensible Firmware Interface (UEFI) firmware. The UEFI firmware is then used to authenticate the boot.efi file within the Intel processor of the TOE host device. The boot.efi file then authenticates the macOS immutable kernel. The macOS then authenticates third party kernel extensions (kexts) and OS user space.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TST_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_TST_EXT.1-AGD-01

There are no AGD evaluation activities for this SFR.

Summary

There are no AGD evaluation activities for this SFR.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TST_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_TST_EXT.1-ATE-01

There are no test evaluation activities for this SFR.

Summary

There are no test evaluation activities for this SFR.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TST_EXT.1-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_TST_EXT.1-AKM-01

There are no KMD evaluation activities for this SFR.

Summary

There are no KMD evaluation activities for this SFR.

2.2.4.6 Trusted Update (FPT_TUD_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TUD_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it describes information stating that an authorized source signs TOE updates and will have an associated digital signature. The evaluator shall examine the TSS contains a definition of an authorized source along with a description of how the TOE uses public keys for the update verification mechanism in the Operational Environment. The evaluator ensures the TSS contains details on the protection and maintenance of the TOE update credentials.

If the Operational Environment performs the signature verification, then the evaluator shall examine the TSS to ensure it describes, for each platform identified in the ST, the interface(s) used by the TOE to invoke this cryptographic functionality.

Summary

The [ST]☝ provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST]☝ is addressed by the TOE. [ST]☝ Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FPT_TUD_EXT.1/AA. A vendor-controlled server is leveraged for obtaining firmware update code packages. The code packages containing the macOS, T2OS/firmware, and sepOS/firmware are all bundled together. The T2OS and sepOS firmware is stored within the Secure Enclave. The TOE stores the incoming update in a temporary location on flash. Once the transfer is complete, the SEP verifies the digital signature using the algorithm described in FCS_COP.1(a). If the verification is successful, the TOE installs the update and reboots the Mac. If the verification is unsuccessful, the TOE terminates the updates process.

Assurance Activity AA-FDEEECPP-FPT_TUD_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it describes information stating that an authorized source signs TOE updates and will have an associated digital signature. The evaluator shall examine the TSS contains a definition of an authorized source along with a description of how the TOE uses public keys for the update verification mechanism in the Operational Environment. The evaluator ensures the TSS contains details on the protection and maintenance of the TOE update credentials.

If the Operational Environment performs the signature verification, then the evaluator shall examine the TSS to ensure it describes, for each platform identified in the ST, the interface(s) used by the TOE to invoke this cryptographic functionality.

Summary

The [ST] provides the TOE summary specification (TSS) in chapter 7, describing how each security functional requirement (SFR) defined in section 6.1 in [ST] is addressed by the TOE. [ST] Table 7 in chapter 7 "TOE Summary Specification" provides for each SFR a detailed description.

The evaluator examined the provided information for FPT_TUD_EXT.1/EE. A vendor-controlled (Apple) server is leveraged for obtaining firmware update code packages. The code packages containing the macOS, T2OS/firmware, and sepOS/firmware are all bundled together. The T2OS and sepOS firmware are stored within the Secure Enclave. The TOE stores the incoming update in a temporary location on flash. Once the transfer is complete, the SEP verifies (authenticates) the digital signature on the bundle using the RTU pubic key and the algorithm described in FPT_FUA_EXT.1. If the verification succeeds, the TOE installs the update and reboots the Mac. If the verification fails, the TOE terminates the updates process with an error message.

Guidance Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TUD_EXT.1-AGD-01

> *The evaluator ensures that the operational guidance describes how the TOE obtains vendor updates to the TOE; the processing associated with verifying the digital signature of the updates (as defined in FCS_COP.1(a)); and the actions that take place for successful and unsuccessful cases.*

Summary

Per section 2.2 of [CCGuide], *Update*, FileVault updates are bundled with macOS updates. Any user can check for updates to FileVault by navigating to Apple menu > About This Mac > Software Update or by opening the System Settings app and selecting Software Update. If updates are available, the user will be given the option of installing updates. Users can also check for updates from the command line by issuing the softwareupdate -l command.

The user must be an authorized user (i.e., successfully logged in) to initiate an update.

Section 2.2 further explains that once the download is complete, the SEP verifies the digital signature on the software update bundle using the RTU public key. If the verification succeeds, the TOE installs the update and reboots the Mac. If the verification fails, the TOE terminates the update process with an error message.

Assurance Activity AA-FDEEECPP-FPT_TUD_EXT.1-AGD-01

> *The evaluator ensures that the operational guidance describes how the TOE obtains vendor updates to the TOE; the processing associated with verifying the digital signature of the updates (as defined in FCS_COP.1(a)); and the actions that take place for successful and unsuccessful cases.*

Summary

Per section 2.2 of [CCGuide], *Update*, FileVault updates are bundled with macOS updates. Any user can check for updates to FileVault by navigating to Apple menu > About This Mac > Software Update or by opening the System Settings app and selecting Software Update. If updates are available, the user will be given the option of installing updates. Users can also check for updates from the command line by issuing the softwareupdate -l command.

The user must be an authorized user (i.e., successfully logged in) to initiate an update.

Section 2.2 further explains that once the download is complete, the SEP verifies the digital signature on the software update bundle using the RTU public key. If the verification succeeds, the TOE installs the update and reboots the Mac. If the verification fails, the TOE terminates the update process with an error message.

Test Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TUD_EXT.1-ATE-01

> *The evaluators shall perform the following tests (if the TOE supports multiple signatures, each using a different hash algorithm, then the evaluator performs tests for different combinations of authentic and unauthentic digital signatures and hashes, as well as for digital signature alone):*
>
> *Test 1: The evaluator performs the version verification activity to determine the current version of the TOE. After the update tests described in the following tests, the evaluator performs this activity again to verify that the version correctly corresponds to that of the update.*
>
> *Test 2: The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that an update successfully installs on the TOE. The evaluator shall perform a subset of other evaluation activity tests to demonstrate that the update functions as expected.*

Summary

Test 1: the evaluator verified that the TOE has a version verification capability. The currently installed version is verified to be macOS "13.2.1 (22D68)".

Test 2: the evaluator verified that the TOE allows for the update of the macOS version.

Assurance Activity AA-FDEEECPP-FPT_TUD_EXT.1-ATE-01

> *The evaluators shall perform the following tests (if the TOE supports multiple signatures, each using a different hash algorithm, then the evaluator performs tests for different combinations of authentic and unauthentic digital signatures and hashes, as well as for digital signature alone):*
>
> *Test 1: The evaluator performs the version verification activity to determine the current version of the TOE. After the update tests described in the following tests, the evaluator performs this activity again to verify that the version correctly corresponds to that of the update.*
>
> *Test 2: The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that an update successfully installs on the TOE. The evaluator shall perform a subset of other evaluation activity tests to demonstrate that the update functions as expected.*

Summary

Test 1: the evaluator verified that the TOE has a version verification capability. The currently installed version is verified to be macOS "13.2.1 (22D68)".

Test 2: the evaluator verified that the TOE allows for the update of the macOS version.

Key Management Assurance Activities

Assurance Activity AA-FDEAACPP-FPT_TUD_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

Assurance Activity AA-FDEEECPP-FPT_TUD_EXT.1-AKM-01

> *There are no KMD evaluation activities for this SFR.*

Summary

There are no KMD evaluation activities for this SFR.

2.3 Security Assurance Requirements

2.3.1 Development (ADV)

2.3.1.1 Basic functional specification (ADV_FSP.1)

Assurance Activity AA-FDEAACPP-ADV_FSP.1-ADV-01

The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

Summary

The developer has provided the following documents as the interface documentation of the TOE:

- [FSPMapping]☝ Apple macOS 13 Ventura: FileVault FSP Mapping,
- [ST]☝ Apple macOS 13 Ventura: FileVault Security Target,
- [CCGuide]☝ Apple macOS 13 Ventura: FileVault Common Criteria Configuration Guide,
- [macOS_UG]☝ macOS User Guide,
- [API_Doc]☝ Apple Developer Documentation.

The [FSPMapping]☝ document lists the TSFIs supported by the TOE. Moreover, the [FSPMapping]☝ document maps each TSFI to multiple SFRs claimed in the [ST]☝. Therefore, the evaluator determined that each TSFI is security relevant.

For each TSFI supported by the TOE, the evaluator examined the interface documentation for the description of that TSFI. The evaluator verified that the interface documentation describes the purpose and method of use for each TSFI.

Assurance Activity AA-FDEAACPP-ADV_FSP.1-ADV-02

The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.

Summary

For each TSFI supported by the TOE, the evaluator examined the interface documentation for the description of that TSFI. The evaluator could see that the parameters are sufficiently explained in the documentation. The evaluator was consistently able to identify all parameters of each TSFI.

Assurance Activity AA-FDEAACPP-ADV_FSP.1-ADV-03

The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

Summary

The [FSPMapping]☝ document lists the TSFIs supported by the TOE and traces them to the SFRs claimed in the [ST]☝. The SFRs claimed in the [ST]☝ consists of both SFRs for Authorization Acquisition (AA) component and SFRs for Encryption Engine (EE) component. The mapping of the interfaces to SFRs for AA component is included and presented the [FSPMapping]☝.

Assurance Activity AA-FDEEECPP-ADV_FSP.1-ADV-01

The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

Summary

The developer has provided the following documents as the interface documentation of the TOE:

- [FSPMapping]☝ Apple macOS 13 Ventura: FileVault FSP Mapping,
- [ST]☝ Apple macOS 13 Ventura: FileVault Security Target,
- [CCGuide]☝ Apple macOS 13 Ventura: FileVault Common Criteria Configuration Guide,
- [macOS_UG]☝ macOS User Guide,
- [API_Doc]☝ Apple Developer Documentation.

The [FSPMapping]☝ document lists the TSFIs supported by the TOE. Moreover, the [FSPMapping]☝ document maps each TSFI to multiple SFRs claimed in the [ST]☝. Therefore, the evaluator determined that each TSFI is security relevant.

For each TSFI supported by the TOE, the evaluator examined the interface documentation for the description of that TSFI. The evaluator verified that the interface documentation describes the purpose and method of use for each TSFI.

Assurance Activity AA-FDEEECPP-ADV_FSP.1-ADV-02

> *The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.*

Summary

For each TSFI supported by the TOE, the evaluator examined the interface documentation for the description of that TSFI. The evaluator could see that the parameters are sufficiently explained in the documentation. The evaluator was consistently able to identify all parameters of each TSFI.

Assurance Activity AA-FDEEECPP-ADV_FSP.1-ADV-03

> *The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.*

Summary

The [FSPMapping]☝ document lists the TSFIs supported by the TOE and traces them to the SFRs claimed in the [ST]☝. The SFRs claimed in the [ST]☝ consists of both SFRs for Authorization Acquisition (AA) component and SFRs for Encryption Engine (EE) component. The mapping of the interfaces to SFRs for EE component is included and presented the [FSPMapping]☝.

2.3.2 Guidance documents (AGD)

2.3.2.1 Operational user guidance (AGD_OPE.1)

Assurance Activity AA-FDEAACPP-AGD_OPE.1-AGD-01

> *The evaluator shall check the requirements below are met by the operational guidance.*
>
> *Operational guidance documentation shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.*
>
> *Operational guidance must be provided for every Operational Environment that the TOE supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target. This may be contained all in one document.*
>
> *The contents of the operational guidance will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in sections 2, 3, and 4 above.*
>
> *In addition to SFR-related Evaluation Activities, the following information is also required.*

> - *The operational guidance shall contain instructions for configuring any 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 TOE will likely contain security functionality that does not fall under the scope of evaluation under this cPP. The operational guidance shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Section 3.2 of [CCGuide]☝, *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state. The low-power sleep state must be disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1.

Section 1.3 of [CCGuide]☝, *Excluded Functionality*, states that the evaluation is limited to the FDE functionality. The following product functionality is not included in the CC evaluation:

- General Purpose Operating System functionality – The TOE is an integral part of macOS 13 Ventura; however, the evaluation is limited to the FDE functionality.
- Disk unlocking using an iCloud account.

Assurance Activity AA-FDEEECPP-AGD_OPE.1-AGD-01

> *The evaluator shall check the requirements below are met by the operational guidance. It should be noted that operational guidance may take the form of an “integrator's guide”, where the TOE developer provides a description of the interface (e.g., commands that the Host Platform may invoke to configure a SED).*
>
> *Operational guidance documentation shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.*
>
> *Operational guidance must be provided for every Operational Environment that the TOE supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target. This may be contained all in one document.*
>
> *The contents of the operational guidance will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in sections 2, 3, and 4 above.*
>
> *In addition to SFR-related Evaluation Activities, the following information is also required.*
>
> - *The operational guidance shall contain instructions for configuring any 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 operational guidance shall describe how to configure the IT environments that are supported to shut down after an administratively defined period of inactivity.*
> - *The operational guidance shall identify system “sleeping” states for all supported operating environments and for each environment, provide administrative guidance on how to disable the sleep state. As stated above, the TOE developer may be providing an integrator's guide and “power states” may be an abstraction that SEDs provide at various levels – e.g., may simply provide a command that the Host Platform issues to manage the state of the device, and the Host Platform is responsible for providing a more sophisticated power management scheme.*
> - *The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The operational guidance shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.*

Summary

Section 1.1 of [CCGuide]☝, *Target of Evaluation*, states that all processing for cryptography related to FDE functionality is performed using the SEP or AES Engine. Apple silicon-based systems use AES-XTS-256 for data encryption and Apple T2-based systems use AES-XTS-128 for data encryption. No configuration of cryptographic engines, algorithms, or key sizes is necessary or available.

Section 3.2 of [CCGuide], *Sleep State*, states that macOS has a low-power sleep state, but the drive is only assumed secure when in a powered-off state. The low-power sleep state can be disabled by having an administrator open Terminal and run sudo pmset -a disablesleep 1.

Section 1.3 of [CCGuide], *Excluded Functionality*, states that the evaluation is limited to the FDE functionality. The following product functionality is not included in the CC evaluation:

- General Purpose Operating System functionality – The TOE is an integral part of macOS 13 Ventura; however, the evaluation is limited to the FDE functionality.
- Disk unlocking using an iCloud account.

2.3.2.2 Preparative procedures (AGD_PRE.1)

Assurance Activity AA-FDEAACPP-AGD_PRE.1-AGD-01

> *The evaluator shall check the requirements below are met by the preparative procedures.*
>
> *The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in section 2 above.*
>
> *Preparative procedures shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.*
>
> *The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in section 2 above.*
>
> *In addition to SFR-related Evaluation Activities, the following information is also required.*
>
> *Preparative procedures must include a description of how the administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target). The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE itself).*
>
> *Preparative procedures must be provided for every Operational Environment that the TOE supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target. This may be contained all in one document.*
>
> *The preparative procedures must include*
>
> - *instructions to successfully install the TSF in each Operational Environment; and*
> - *instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and*
> - *instructions to provide a protected administrative capability.*

Summary

Section 2.1 of [CCGuide] describes how to install the TOE. Installation is the same for all platforms. The TOE operational environments are the platforms listed in Table 2 of [CCGuide].

Section 1.1 of [CCGuide] describes how the platform provides security functionality for the TOE.

Configuration and Management of the TOE is described in section 3 of [CCGuide], including

- enable encryption
- sleep state
- add/delete user
- change passwords
- delete encryption keys
- configuring firewall

[CCGuide] is written in a clear and reasonable mannner such that it can easily be understood by the target audience and is consistent with [ST].

For example, [CCGuide] Table 2 lists all the hardward platforms. Table 8 of [ST] also lists the hardware platforms and is consistent with [CCGuide] Table 2.

Assurance Activity AA-FDEEECPP-AGD_PRE.1-AGD-01

The evaluator shall check the requirements below are met by the preparative procedures.

The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in sections 2, 3, and 4 above.

Preparative procedures shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in sections 2, 3, and 4 above.

In addition to SFR-related Evaluation Activities, the following information is also required.

Preparative procedures must include a description of how the administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target). The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE itself).

Preparative procedures must be provided for every Operational Environment that the TOE supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target. . This may be contained all in one document.

The preparative procedures must include

- *instructions to successfully install the TSF in each Operational Environment; and*
- *instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and*
- *instructions to provide a protected administrative capability.*

Summary

Section 2.1 of [CCGuide] describes how to install the TOE. Installation is the same for all platforms. The TOE operational environments are the platforms listed in Table 2 of [CCGuide].

Section 1.1 of [CCGuide] describes how the platform provides security functionality for the TOE.

Configuration and Management of the TOE is described in section 3 of [CCGuide], including

- enable encryption
- sleep state
- add/delete user
- change passwords
- delete encryption keys
- configuring firewall

[CCGuide] is written in a clear and reasonable mannner such that it can easily be understood by the target audience and is consistent with [ST].

For example, [CCGuide] Table 2 lists all the hardward platforms. Table 8 of [ST] also lists the hardware platforms and is consistent with [CCGuide] Table 2.

2.3.3 Tests (ATE)

2.3.3.1 Independent testing - conformance (ATE_IND.1)

Assurance Activity AA-FDEAACPP-ATE_IND.1-ATE-01

The evaluator shall examine the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Summary

The test environment was set up according to a setup strategy that followed the evaluated configuration requirements specified in the guidance, supplemented by configurations required to perform testing. The test configuration at the atsec location is made up of the TOEs connected to a private WLAN network, which itself is connected to the internet. At the Apple location, the test configuration consists of the TOEs connected using a proprietary cable to a test laptop.

In particular, Disk Utility version 22.3 was used for the testing.

Assurance Activity AA-FDEAACPP-ATE_IND.1-ATE-02

The evaluator shall examine the TOE to determine that it has been installed properly and is in a known state.

Summary

The evaluator verified that the TOE was installed properly, running a production version of the macOS 13.2.1 (22D68) operating system.

Assurance Activity AA-FDEAACPP-ATE_IND.1-ATE-03

The evaluator shall prepare a test plan that covers all of the testing actions for ATE_IND.1 in the CEM and in the SFR-related Evaluation Activities. While it is not necessary to have one test case per test listed in an Evaluation Activity, the evaluator must show in the test plan that each applicable testing requirement in the SFR-related Evaluation Activities is covered.

The test plan identifies the platforms to be tested, and for any 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 and configuration of each platform to be tested, and any setup actions that are 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 any cryptographic engine to be used (e.g. for cryptographic protocols being evaluated).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives, and the expected results.

The test report (which could just be an updated 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, so that a fix was then installed and then a successful re-run of the test was carried out, then the report would show a "fail" result followed by a "pass" result (and the supporting details), and not just the "pass" result.

Summary

The Detailed Test Report [DTR]☝ specifies all the tests covering the assurance activities from [FDE_AA]☝ and [FDE_EE]☝. The [DTR]☝ is provided to the certification body but is not published.

Description of the test configuration is provided in AA-FDEEECPP-ATE_IND.1-ATE-03.

Assurance Activity AA-FDEEECPP-ATE_IND.1-ATE-01

The evaluator shall examine the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Summary

The test environment was set up according to a setup strategy that followed the evaluated configuration requirements specified in the guidance, supplemented by configurations required to perform testing. The test configuration at the atsec location is made up of the TOEs connected to a private WLAN network, which itself is connected to the internet. At the Apple location, the test configuration consists of the TOEs connected using a proprietary cable to a test laptop.

In particular, Disk Utility version 22.3 was used for the testing.

Assurance Activity AA-FDEEECPP-ATE_IND.1-ATE-02

> *The evaluator shall examine the TOE to determine that it has been installed properly and is in a known state.*

Summary

The evaluator verified that the TOE was installed properly, running a production version of the macOS 13.2.1 (22D68) operating system.

Assurance Activity AA-FDEEECPP-ATE_IND.1-ATE-03

> *The evaluator shall prepare a test plan that covers all of the testing actions for ATE_IND.1 in the CEM and in the SFR-related Evaluation Activities. While it is not necessary to have one test case per test listed in an Evaluation Activity, the evaluator must show in the test plan that each applicable testing requirement in the SFR-related Evaluation Activities is covered.*
>
> *The test plan identifies the platforms to be tested, and for any 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 and configuration of each platform to be tested, and any setup actions that are 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 any cryptographic engine to be used (e.g. for cryptographic protocols being evaluated).*
>
> *The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives, and the expected results.*
>
> *The test report (which could just be an updated 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, so that a fix was then installed and then a successful re-run of the test was carried out, then the report would show a “fail” result followed by a “pass” result (and the supporting details), and not just the “pass” result.*

Summary

The Detailed Test Report [DTR]☝ specifies all the tests covering the assurance activities from [FDE_AA]☝ and [FDE_EE]☝. The [DTR]☝ is provided to the certification body but is not published.

The test environment was set up according to a setup strategy that followed the evaluated configuration requirements specified in the guidance, supplemented by configurations required to perform testing. The test configuration is made up of the TOE connected to a private WLAN network which also hosts a Linux system and a macOS Server. The macOS Server is created on a macOS system upon which the macOS Server software, available from the Apple app store, is installed.

The test platforms are outlined below, which are derived from the devices claimed in [ST]:

- Intel Xeon W iMac Pro (iMacPro1,1)

- M1 Mac mini (Macmini9,1)
- M2 Mac mini (Mac14,3)

All plaforms were time synchronized using the Apple NTP server (time.apple.com). The tested operating system version is macOS 13.2.1.

The following tools were used for testing: Apple tools

- MacBook Pro (MacBookPro18,2) laptop used by the vendor to interface with the TOE platforms. This MacBook also uses the Apple NTP server.
- Proprietary specialized Apple USB-C cable
- TOE platforms listed above running development-fused (dev-fused) macOS version 13.2.1.
- Apple internal software

atsec tools

- TOE platforms listed above running macOS 13.2.1.
- Disk Utility version 22.3

The following diagram shows the test configurations used.

Figure 1: CCTL test configuration

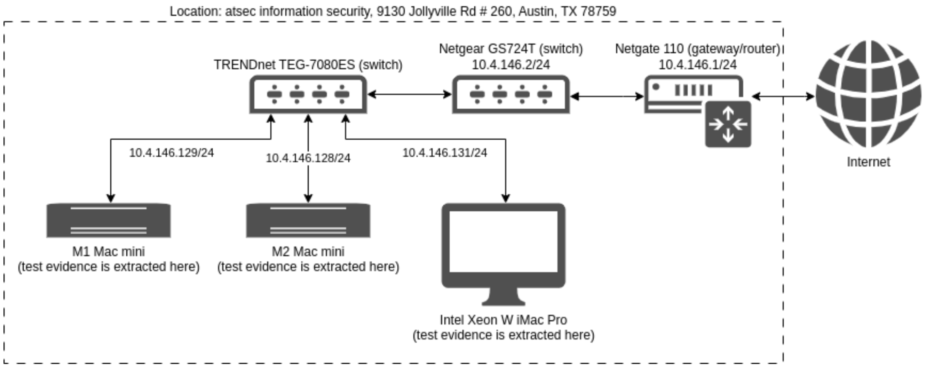
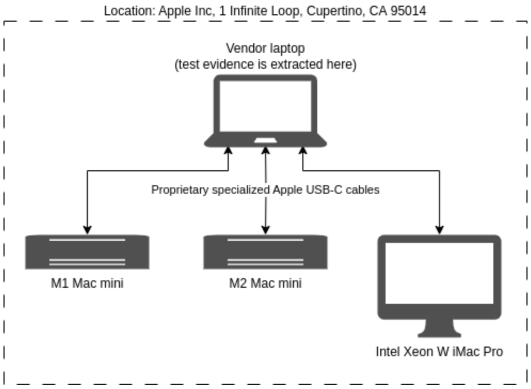

Location: atsec information security, 9130 Jollyville Rd # 260, Austin, TX 78759
TRENDnet TEG-7080ES (switch)
Netgear GS724T (switch)
10.4.146.2/24
Netgate 110 (gateway/router)
10.4.146.1/24
Internet
10.4.146.129/24
10.4.146.128/24
10.4.146.131/24
M1 Mac mini
(test evidence is extracted here)
M2 Mac mini
(test evidence is extracted here)
Intel Xeon W iMac Pro
(test evidence is extracted here)

Figure 2: Apple test configuration


Location: Apple Inc, 1 Infinite Loop, Cupertino, CA 95014
Vendor laptop
(test evidence is extracted here)
Proprietary specialized Apple USB-C cables
M1 Mac mini
M2 Mac mini
Intel Xeon W iMac Pro

2.3.4 Vulnerability assessment (AVA)

2.3.4.1 Vulnerability Survey (AVA_VAN.1)

Assurance Activity AA-FDEAACPP-AVA_VAN.1-AVA-01

> *The evaluator shall examine the documentation outlined below provided by the vendor to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.*
>
> *In addition to the activities specified by the CEM in accordance with Table 2 above, the evaluator shall perform the following activities.*

Summary

The evaluator found in the [ST]☝ the information about TOE version both software and hardware, including libraries used by the TOE which are reported below.

Table 8: TOE software components identification

Component	Version
Apple macOS	13.2.1
FileVault	2
Apple corecrypto Module	13.0

The evaluator's assessment cover all the platforms specified in the ST. In addition to these platforms the evaluator found also Apple DMA controller 1.0 hardware. The evaluator examined the documentation provided by the vendor and confirmed that it contained all the required information.

Assurance Activity AA-FDEAACPP-AVA_VAN.1-AVA-02

> *The evaluator formulates hypotheses in accordance with process defined in [cPP] Appendix A.1. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in [cPP] Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with [cPP] Appendix A.2. The results of the analysis shall be documented in the report according to [cPP] Appendix A.3.*

Summary

The evaluator searched for publicly known vulnerabilities applicable to Apple macOS 13 Venture: FileVault using the following sources.

- Apple security content disclosure statements for releases of Apple macOS 13 Venture: FileVault related to this evaluation:
  - https://support.apple.com/en-us/HT213633 macOS Ventura 13.2.1 Released February 13, 2023
  - https://support.apple.com/en-us/HT213670 macOS Ventura 13.3 Released March 27, 2023
  - https://support.apple.com/en-us/HT213721 macOS Ventura 13.3.1 Released April 7, 2023
  - https://support.apple.com/en-us/HT213758 macOS Ventura 13.4 Released May 18, 2023
  - https://support.apple.com/en-us/HT213813 macOS Ventura 13.4.1 Released June 21, 2023
  - https://support.apple.com/en-us/HT213825 macOS Ventura 13.4.1 Rapid Security Responses (a) and (c) Released July 10, 2023
  - https://support.apple.com/en-us/HT213843 macOS Ventura 13.5 Released July 24, 2023
  - https://support.apple.com/en-us/HT213906 macOS Ventura 13.5.2 Released September 7, 2023

  - https://support.apple.com/en-us/HT213931 macOS Ventura 13.6 Released September 21, 2023
  - https://support.apple.com/en-us/HT213985 macOS Ventura 13.6.1 Released October 25, 2023
- MITRE Common Vulnerabilities and Exposures (CVE) List:
  - http://cve.mitre.org/cve/
- Search the National Vulnerability Database:
  - https://nvd.nist.gov/
- Search US-CERT:
  - http://www.kb.cert.org/vuls/html/search
- Google:
  - https://google.com
- Apple security updates:
  - https://support.apple.com/en-us/HT201222

The following search terms were used on the MITRE, NIST, US-CERT and Google web sites: As required by the [FDE_AA]☝ and [FDE_EE]☝ PPs the evaluator will consider the following search terms:

Table 9: Search Terms

PPs required Search Term	Keyword	Description
Product name	Apple macOS 13 Ventura: FileVault	ST 1.2 TOE Identification
Underlying components (e.g., OS, software libraries (crypto libraries), chipsets)	macos Ventura 13	The evaluator identified the keyword in the [ST]☝ according to the PP's required search terms.
Drive encryption, disk encryption	AES used in XTS mode	The ST in 6.1.2.1 Protection of Data on Disk (FDP_DSK_EXT.1) state that the Full Drive Encryption is performed in accordance with FCS_COP.1(f) that claims AES XTS mode for this purpose.
key destruction, key sanitization	single overwrite zeroization	The ST in 6.1.1.5 FCS_CKM.4(a)/AA Cryptographic Key Destruction (Power Management) and 6.1.1.6 FCS_CKM.4(a)/EE Cryptographic Key Destruction (Power Management) - Encryption Engine explains how the TOE supports key destruction for both AA and EE. The evaluator identified the keyword according to the zeroization method used by the TOE.
Opal management software, SED management software	none.	The evaluator searches in the ST the terms: • OPAL • SED • Management • management software

PPs required Search Term	Keyword	Description
		and found that the SED management software used is essentially the macOS operating system, which is already a keyword used in the search. No OPAL storage specification were identified in the ST.
Password caching and Key caching	None.	The evaluator searches in the ST the terms: • Password caching • Key caching • cache • caching • Key • Password and found that the Keys and Password derivation materials are stored in volatile memory while being used for specific operation. The TOE destroy all key material when transitioning to a Compliant power saving state or once the keys are no longer required, trough single overwrite zeroization. Therefore no new search terms were identified.
Underlying components (e.g., OS, software libraries (crypto libraries), chipsets)	Apple corecrypto Module 13.0 (Software and Hardware)	The evaluator identified the keyword in the [ST]☝ according to the PP's required search terms.
Underlying components (e.g., OS, software libraries (crypto libraries), chipsets)	Apple silicon Intel with T2 Secure Enclave Amber Lake Cascade Lake Coffee Lake Comet Lake Ice Lake Skylake ARM 8.5 ARM 8.6 Core i5-1030NG7 Core i5-1038NG7 Core i5-10500 Core i5-10600 Core i5-8210Y Core i5-8257U Core i5-8259U Core i5-8279U Core i5-8500 Core i5-8500B Core i5-8557U Core i5-8600 Core i5-9600K	The evaluator identified the keyword in the [ST]☝ according to the PP's required search terms.

PPs required Search Term	Keyword	Description
	Core i7-1060NG7 Core i7-1068NG7 Core i7-10700K Core i7-8557U Core i7-8559U Core i7-8569U Core i7-8700 Core i7-8700B Core i7-8750H Core i7-8850H Core i7-9750H Core i9-10910 Core i9-8950HK Core i9-9880H Core i9-9900K Core i9-9980HK Xeon W-2140B Xeon W-2150B Xeon W-2170B Xeon W-2191B Xeon W-3223 Xeon W-3235 Xeon W-3245 Xeon W-3265M Xeon W-3275M	

In addition to the lists of fixes published by the vendor, the evaluator performed manual searches on multiple occasions between 2023-05-08 and 2023-05-12, 2023-05-18 and 2023-05-22, 2023-09-29 and 2023-10-03, 2023-10-25 and 2023-10-26, 2023-11-02, and 2023-11-28. The developer publishes security content disclosure statements providing information about vulnerabilities fixed in each release of macOS after 13.2.1. The developer also provides continuous updates to the TOE, therefore the TOE version tested in this evaluation, macOS 13.2.1 Ventura, is no longer available. As of the date of this report, the latest version relevant to this evaluation is macOS 13.6.1. The evaluator's CVE search found no vulnerabilities apart from the ones listed in the security content disclosure statements, all of which have been fixed in subsequent releases of macOS 13.

Assurance Activity AA-FDEEECPP-AVA_VAN.1-AVA-01

> *The evaluator formulates hypotheses in accordance with process defined in Appendix A.1. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.*

Summary

This work unit has been worked in conjuction with AA-FDEAACPP-AVA_VAN.1-AVA-02

A Appendixes

A.1 References

API_Doc
Apple Developer Documentation
Author(s) Apple Inc.
Date received July 2023
Location https://developer.apple.com/documentation/technologies

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-LG
CCEVS LabGrams
Author(s) NIAP
Date November 2023
Location https://www.niap-ccevs.org/Documents_and_Guidance/labgrams.cfm

CCEVS-PL
CCEVS Scheme Policy Letters
Author(s) NIAP
Date November 2023
Location https://www.niap-ccevs.org/Documents_and_Guidance/policy.cfm

CCEVS-PUB
CCEVS Scheme Publications
Author(s) NIAP
Date November 2023
Location https://www.niap-ccevs.org/Documents_and_Guidance/guidance_docs.cfm

CCEVS-TD
Technical Decisions
Author(s) NIAP
Date November 2023
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_tds.cfm

CCGuide
Apple macOS 13 Ventura: FileVault Common Criteria Configuration Guide
Version 1.1
Date 2023-11-28
File name agd/vid11348_filevault_cc_guide_v1.1.pdf

CEM
Common Methodology for Information Technology Security Evaluation

Version 3.1R5
Date April 2017
Location http://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf

DTR **Apple macOS 13 Ventura: FileVault Detailed Test Report**
Date 2023-11-28
File name ase/macOS_13_FileVault-DTR.v1.1.pdf

FDE_AA **collaborative Protection Profile for Full Drive Encryption - Authorization Acquisition**
Version 2.0 + Errata 20190201
Date 2019-02-01
Location https://www.niap-ccevs.org/MMO/PP/CPP_FDE_AA_V2.0E.pdf

FDE_EE **collaborative Protection Profile for Full Drive Encryption - Encryption Engine**
Version 2.0 + Errata 20190201
Date 2019-02-01
Location https://www.niap-ccevs.org/MMO/PP/CPP_FDE_EE_V2.0E.pdf

FSPMapping **Apple macOS 13 Ventura: FileVault FSP Mapping**
Version 1.1
Date 2023-06-30
File name adv/apple_filevault_fsp_mapping.xlsx

KMD **Apple macOS 13 Ventura: FileVault Key Management Description (KMD)**
Version 1.0
Date 2023-11-29
File name akm/filevault-kmd_v1.1.pdf

macOS_UG **macOS User Guide**
Author(s) Apple Inc.
Version macOS 13
Date received July 2023
Location https://support.apple.com/guide/mac-help

ST **Apple macOS 13 Ventura: FileVault Security Target**
Version 1.1
Date 2023-11-28
File name ase/st-macOS13-filevault-v1.1.pdf