Specification of Functional Requirements for Cryptography

Maintained by: CCDB

Unique Identifier: 018

Version: 1.0

Status: Final

Date of issue: 2025-01-31

Approved by: CWG

Contents

Tables

1. Introduction

The Common Criteria Development Board tasked the Cryptographic Working Group with creating a catalog of cryptographic components in order to harmonize use of the FCS Class across all Common Criteria (CC) Requirements Documents recognized by Common Criteria Recognition Agreement (CCRA) schemes.

1.1. Purpose

This document provides a set of cryptographic components based on the FCS Class of CC:2022 Revision1 along with guidelines on how to incorporate the components into CC Requirements Documents such as Protection Profiles, collaborative Protection Profiles, Protection Profile Modules, Protection Profile Configuration, Functional Packages, and Security Targets.

Many of the recommendations include alternative selections that attempt to represent the various algorithms, parameters, and standards that are acceptable to at least one CCRA scheme.

1.2. Intended Audience

This document is intended to provide guidance to technical communities engaged in the development of CC Requirements Documents.

1.3. Common Criteria Documents

The components in this document are based on or derived from those in CC:2022 Revision 1:

Common Criteria for Information Technology Security Evaluation

- *Part 1: Introduction and general model*, CCMB-2022-11-001, Nov 2022, CC:2022 Revision 1.
- *Part 2: Security functional components*, CCMB-2022-11-002, Nov 2022, CC:2022 Revision 1.
- *Part 3: Security assurance components*, CCMB-2022-11-003, Nov 2022, CC:2022 Revision 1.
- *Part 4: Framework for the specification of evaluation methods and activities*, CCMB-2022-11-004, Nov 2022, CC:2022 Revision 1.
- *Part 5: Pre-defined packages of security components*, CCMB-2022-11-005, Nov 2022, CC:2022 Revision 1.

Common Methodology for Information Technology Security Evaluation

- *Evaluation methodology*, CCMB-2022-11-006, Nov 2022, CC:2022 Revision 1.

Errata for CC:2022 (Release 1), parts 1 to 5, and CEM:2022 (Release 1) providing appropriate solutions as proposed corrections or interpretations, respectively.

- *Errata and Interpretation for CC:2022 (Release 1) and CEM: 2022 (Release 1)*, CCMB-2024-07-002 Version 1.1, July 2024.

This catalog includes two supplemental documents. Evaluation Activities for the components in the catalog will be specified in a separate Evaluation Methods document to be published in the near future:

- *Evaluation Methods for Cryptographic Security Functional Requirements*, and

A glossary containing definitions of terms referenced in the catalog appears in a separate document.

- *Supporting Document Guidance: Cryptographic Definitions*.

2. Overview

2.1. Organization of this Document

This document is organized similarly to CC:2022 Revision 1 Part 2. Components are organized by class and family. In this document, all components are in Class FCS: Cryptographic support.

Components appear alphabetically in the main body of the catalog.

Several of the components in the catalog are extensions of CC:2022 components. The Extended Component Definitions for these components can be found in Annex A: Extended Component Definitions

2.2. How to Use This Document

Requirements Document Authors should be able to copy components directly from the catalog into a Requirements Document. Likewise, Extended Component Definition Information can be copied directly from the ECD Annex into the ECD section of a Requirements Document.

The catalog contains two kinds of Notes. Application Notes provide guidance for ST Authors (users of the Requirements Document). Catalog Guidance Notes provide guidance for Requirements Documents Authors (users of the catalog).

Application Notes contain guidance for ST Authors on how to make selections and assignments when claiming conformance to the published Requirements Document. Application Notes appear after the text of each component and are intended to be carried forward into the Requirements Document.

Catalog Guidance Notes contain guidance to help Requirements Document Authors choose which components to include and which selections to allow. These notes appear before each component in the catalog and should not be carried forward in the published Requirements Document.

For many components in the catalog, selections are grouped together as rows of a table. Many of the requirements include alternative selections that attempt to represent the various algorithms, parameters, and standards that are acceptable to at least one CCRA scheme. Requirements Document Authors should select the rows that apply to their technology and leave out the others. They may also choose to remove selections within the rows that do not apply to the target technology, such as key sizes that products should not support. Likewise, Requirements Document Authors may add components or selections that are not included in the catalog.

Evaluation Activities are specified in the accompanying Evaluation Methods document. Requirements Documents may refer the appropriate sections of the Evaluation Methods document rather than copy the activities into their documents.

2.3. A Note About Dependencies

Some components in this catalog contain a different set of dependencies than those that appear in Part 2 of CC:2022. Some of the differences are due to application of the errata to Part 2. This catalog also replaces some Part 2 components with extended components and adds new extended components. These new extended components have their own specific lists of dependencies on other components—either defined in CC:2022 or elsewhere in this catalog. When using this catalog for the development of Requirements Documents, these dependencies should be considered.

Dependencies for non-extended catalog components should be copied directly from the catalog into the component in the Requirements Document.

Dependencies for extended catalog components may also be copied from the catalog into the Requirements Document, but the dependencies must at least be copied from the catalog's ECD Annex into the ECD section of the Requirements Document along with the other ECD information.

2.4. Typographical Conventions

Keywords in components are in **boldface**. Keywords are

- **selection:**
- **selection, choose one of:**
- **assignment:**
- **refinement**

Contents of selections are in a normal typeface if they are literal values. For example:

> [**selection:** 128, 256] bits

The contents of assignments are in italics since they are not literal values but rather descriptions of the permitted values for the assignment:

> [**assignment:** *numeric value between 1 and 5*]

Text that represents literal completion of an assignment or selection is presented within square brackets and in a normal typeface.

> The TSF shall perform [symmetric key encryption/decryption] in ....

When referenced in Application Notes, literal selection values are surrounded by quotes.

For components that use tables, the text of the requirement contains selections that refer to columns of the table. These selections contain only a single choice that is italicized to indicate that it is not literal, but rather that it refers to a column of the table. The PP/ST author chooses one or more rows in the table which implicitly includes selections in each of the column of the row.

For example:

FCS_COP.1.1/SKC The TSF shall perform [symmetric-key encryption/decryption] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the allowed choices for completion of the selection operations of FCS_COP.1.1/SKC.

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AES-CBC	AES in CBC mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
XTS-AES	AES in XTS mode with unique tweak values that are consecutive non-negative integers starting at an arbitrary non-negative integer	[**selection**: 256, 512] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: IEEE Std. 1619-2018, NIST SP 800-38E] [XTS]
AES-CTR	AES in Counter Mode with a non-repeating initial counter and with no repeated use of counter values across multiple messages with the same secret key.	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] **[selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]
CAM-CBC	Camellia in CBC mode with non-repeating and unpredictable IVs	**[selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]

The Identifier column is referenced by the text of some components. In other components the column is merely there to serve as a convenient shorthand for referring to the table rows.

2.5. Modification of Components

Use of the components in this catalog is strongly encouraged, but not mandated. Requirements Documents authors and schemes are the ultimate decisionmakers regarding the requirements that are appropriate for their target technologies. This catalog provides a framework for the definition and use of cryptographic requirements, but also provides flexibility within that framework such that it should not be necessary to define requirements outside of this framework. Nevertheless,

Requirements Documents Authors are allowed to create new extended components should this catalog not meet the needs of their technology type.

For example, in the above table, if a Technical Community finds that XTS-AES is not appropriate for their technology, the offending row can simply be removed from the table and the remainder of the table be copied into the requirements document.

Likewise, if 192-bit encryption is not appropriate, that choice can be removed from the selections.

If there is another algorithm that can be used for Symmetric-Key Cryptography that is not in the table, another row can be added to the table.

Requirements Document authors should keep in mind that discarding or modifying catalog components may require changes to Dependencies, Extended Component Definitions, or Evaluation Activities.

3. Cryptographic Key Management (FCS_CKM)

3.1. Catalog Guidance Notes for Family FCS_CKM

SFRs under FCS_CKM pertain to cryptographic keys. This includes key management activities that occur during the typical lifecycle of a key. This section includes key generation, key derivation, key distribution, key agreement, key access, and key destruction.

3.1.1. Key Generation and Key Derivation

This catalog distinguishes key generation from key derivation. *Key generation* refers to those instances in which a new key is created from a source of entropy. Those instances in which a reproducible process derives a key from other material that are themselves reasonable sources of entropy are referred to as *key derivation*. Ideally, the sources of entropy in a key derivation process are unknown. However, *password-based key derivation*, which uses low-entropy sources of derivation material that may be easily guessable, has been used and supported for decades. This catalog recommends FCS_CKM.5 Cryptographic Key Derivation for instances in which the sources for derivation are reasonably expected to be unknown and unguessable and introduces FCS_CKM_EXT.8 Password-Based Key Derivation to add constraints, work, or more entropy for the instances in which one or more components of the derivation material may contain limited entropy.

3.1.2. Key Establishment, Key Distribution/Transport, and Key Agreement

NIST SP 800-56A Revision 3 explains that "[a] key-establishment scheme can be characterized as either a key-agreement scheme or a key-transport scheme." *Key agreement* schemes refer to cases in which two or more parties want to establish a single key between them, and all parties contribute to the entropy of the agreed-upon key. *Key transport* schemes refer to cases in which one party has a key to share with another party. In this case, only one party has contributed to the entropy of the key. Since FCS_CKM.2 supports key distribution, this catalog recommends using FCS_CKM.2 Cryptographic Key Distribution to specify key transport schemes and introduces FCS_CKM_EXT.7 Cryptographic Key Agreement to cover key agreement schemes.

NIST SP 800-56A Revision 3, Section 6, presents several key agreement schemes. Rather than list all of them here, this document presents all the primitives necessary to build these schemes. Namely, find Finite Field Cryptography Diffie-Hellman (FFC DH) and Elliptic Curve Diffie-Hellman (ECDH) in FCS_CKM_EXT.7, key derivation functions in FCS_CKM.5, and pseudo-random functions (PRFs) in FCS_COP.1/CMAC, FCS_COP.1/Hash, FCS_COP.1/KeyedHash, and FCS_COP.1/SKC. For integrated encryption schemes such as the Elliptic Curve Integrated Encryption Scheme (ECIES), consult the ECIES standards such as those from ISO, IEEE, ANSI, and SECG. Each has slight variations, but the key agreement primitives can be found in FCS_CKM_EXT.7 and the KDF primitives in FCS_CKM.5.

3.1.3. Key Access

Cryptographic key access applies primarily to the storage of keys for future use and retrieval of keys for immediate use by the TOE. The end goal here is to protect the confidentiality and authenticity of the keys while in storage.

Cryptographic key archival, backup, and escrow – TOEs often perform cryptographic key archival to manage limited memory resources inside their own security boundaries. Often the TOE encrypts the cryptographic keys prior to saving them to storage that is close by, on the same device as the TOE and can be accessed quickly. TOEs may perform cryptographic key backups into storage that is meant for longer term keeping. Backup storage is often not on the same device as the TOE and may be physically hundreds of miles away. In practice, the TOE encrypts the keys using an approved method before sending them to backup storage. TOEs may perform cryptographic key escrow in which it entrusts a third party with access to the private or secret keys. In practice, the TOE protects the keys using a cryptographic key access method agreed upon with the escrow agent before sending them.

Cryptographic key recovery – This refers to the retrieval of cryptographic keys from either archival, backup, or escrow locations. In each case, the TOE uses the agreed upon cryptographic key access method.

3.2. FCS_CKM.1/AKG Cryptographic Key Generation

Catalog Guidance Notes

FIPS PUB 186-5 does not approve Finite Field Cryptography (FFC) DSA for digital signature generation but allows DSA for digital signature verification for legacy purposes. Since it is not approved for digital signature, then methods for key generation are restricted to key agreement.

If the Requirements Document does not include “DH” in FCS_CKM_EXT.7, then it need not include “FCC-ERB” or “FCC-RS” here.

If the Requirements Document includes “ECDH” or “ECDH-Ed” in FCS_CKM_EXT.7, then “ECC–ERB” or “ECC–RS” must be included here.

If the Requirements Document includes “ECDSA” or “EC-KCDSA” in FCS_COP.1/SigGen, then “ECC–ERB” or “ECC–RS” must be included here.

If the Requirements Document includes “EdDSA” in FCS_COP.1/SigGen, then “EdDSA” must be included here.

If the Requirements Document includes “LMS”, “HSS”, “XMSS”, or “XMSS^{MT}” in FCS__COP.1/SigGen, then “LMS”, “HSS”, “XMSS”, or “XMSS^{MT}” must be included here, respectively.

FCS_CKM.1/AKG Cryptographic Key Generation – Asymmetric Key

FCS_CKM.1/AKG	**Asymmetric Cryptographic Key Generation (AKG)**

Hierarchical to: No other components.
Dependencies: [FCS_CKM.2 Cryptographic key distribution, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_COP.1 Cryptographic operation]
[FCS_RBG.1 Random bit generation, or
FCS_RNG.1 Generation of random numbers]
FCS_CKM.6 Timing and event of cryptographic key destruction

FCS_CKM.1.1/AKG The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [**selection**: *cryptographic key generation algorithm*] and specified cryptographic **algorithm parameters** ~~key sizes~~ [**selection**: *cryptographic algorithm parameters*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_CKM.1/AKG:

Table 1: Recommended choices for FCS_CKM.1/AKG

Identifier	**Cryptographic Key Generation Algorithm**	**Cryptographic Algorithm Parameters**	**List of Standards**
RSA	RSA	Modulus of size [**selection**: 2048, 3072, 4096] bits	NIST FIPS PUB 186-5 (Section A.1.1)
ECC-ERB	ECC - Extra Random Bits	Elliptic Curve [**selection**: P-256, brainpoolP256r1, P-384, brainpoolP384r1, P-521, brainpoolP512r1]	NIST FIPS PUB 186-5 (Section A.2.1) [**selection**: NIST SP 800-186 (Section 3) [NIST Curves], RFC 5639 (Section 3) [Brainpool curves]]
ECC-RS	ECC - Rejection Sampling	Elliptic Curve [**selection**: P-256, brainpoolP256r1, P-384, brainpoolP384r1, P-521, brainpoolP512r1]	NIST FIPS PUB 186-5 (Section A.2.2) [**selection**: NIST SP 800-186 (Section 3) [NIST Curves], RFC 5639 (Section 3) [Brainpool curves]]

FFC-ERB	FFC – Extra Random Bits	Static domain parameters approved for [**selection**: IKE groups [**selection**: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection**: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]	NIST SP 800-56A Revision 3 (Section 5.6.1.1.3) [key pair generation] [**selection**: RFC 3526 [IKE groups], RFC 7919 [TLS groups]]
FFC-RS	FFC – Rejection Sampling	Static domain parameters approved for [**selection**: IKE groups [**selection**: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection**: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]	NIST SP 800-56A Revision 3 (Section 5.6.1.1.4) [key pair generation] [**selection**: RFC 3526 [IKE groups], RFC 7919 [TLS groups]]
EdDSA	EdDSA	Domain parameters approved for elliptic curves [**selection:** Edwards25519, Edwards448]	NIST FIPS PUB 186-5 (Section 6.2.1) [key-pair generation] NIST SP 800-186 (Section 3.2.3) [Edwards Curves]
KCDSA	KCDSA	Domain parameters generation with (L, N) = [**selection**: (2048, 224), (2048, 256), (3072, 256)] bits	ISO/IEC 14888-3:2018 (Subclause 6.3) [KCDSA]
EC-KCDSA	EC-KCDSA	Elliptic Curves [**selection**: P-224, B-233, K-233, P-256, B-283, K-283]	ISO/IEC 14888-3:2018 (Subclause 6.7) [EC-KCDSA] NIST SP 800-186 (Section 3) [NIST Curves]
LMS	LMS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]], Winternitz parameter = [**selection**: 1, 2, 4, 8], and tree height = [**selection**: 5, 10, 15, 20, 25]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
HSS	HSS	Private key size = [**selection:** 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]], Winternitz parameter = [**selection**: 1, 2, 4, 8], tree height = [**selection**: 5, 10, 15, 20, 25], and number of levels = [**selection**: 1, 2, 3, 4, 5, 6, 7, 8]	RFC 8554 [HSS] NIST SP 800-208 [parameters]
XMSS	XMSS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]], tree height = [**selection**: 10, 16, 20]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]

XMSS^{MT}	XMSS^{MT}	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]], (total tree height, number of levels) = [**selection**: (20, 2), (20, 4), (40, 2), (40, 4), (40, 8), (60, 3), (60, 6), (60, 12)]	RFC 8391 [XMSS^{MT}] NIST SP 800-208 [parameters]

Application Notes:
For RSA the choice of the modulus implies the resulting key sizes of the public and private keys generated using the specified standard methods.

For Finite Field Cryptography (FFC) DSA, ST authors should consult schemes for guidelines on use. FIPS PUB 186-5 does not approve DSA for digital signature generation but allows DSA for digital signature verification for legacy purposes. "FFC-ERB" or "FFC–RS" may be claimed only for generating private and public keys when "DH" is claimed in FCS_CKM_EXT.7.

When generating ECC keys pairs for key agreement and if "ECDH" or "ECDH-Ed" is claimed in FCS_CKM_EXT.7, then "ECC–ERB" or "ECC–RS" must be claimed. The sizes of the private key, which is a scalar, and the public key, which is a point on the elliptic curve, are determined by the choice of the curve.

When generating ECC key pairs for digital signature generation and if "ECDSA" or "EC-KCDSA" are claimed in FCS_COP.1/SigGen, then "ECC–ERB" or "ECC–RS" must be claimed. The sizes of the private key, which is a scalar, and the public key, which is a point on the elliptic curve, are determined by the choice of the curve.

When generating EdDSA key pairs for digital signatures and if "EdDSA" is claimed in FCS_COP.1/SigGen, then "EdDSA" must be claimed here. The chosen domain parameters determine the size of the private keys and the public keys.

For LMS, HSS, XMSS, and XMSS^{MT}, the key sizes do not represent the expected security strength. All key sizes given here correspond to an expected security strength of 128 bits, per NIST SP 800-208.

For HSS and XMSS^{MT} the same hash or XOF function must be used at each level. Within each level, the same Winternitz parameter must be used but can be different for each level. For HSS, within each level, the same tree height must be used but can be different for each level.

3.3. FCS_CKM.1/SKG Cryptographic Key Generation – Symmetric Key

Catalog Guidance Notes

Include this component if the TOE supports creating symmetric keys directly from the output of an RBG without further conditioning.

To derive symmetric keys from other keying material, see FCS_CKM.5. To derive symmetric keys from passwords, see FCS_CKM_EXT.8. To derive symmetric keys from keying material contributed from two parties, see FCS_CKM_EXT.7.

FCS_CKM.1/SKG Cryptographic Key Generation – Symmetric Key

FCS_CKM.1/SKG	Cryptographic Key Generation – Symmetric Key (SKG)

Hierarchical to: No other components.
Dependencies: [FCS_CKM.2 Cryptographic key distribution, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic Key Agreement, or
FCS_COP.1 Cryptographic operation]
FCS_CKM.6 Timing and event of cryptographic key destruction
[FCS_RBG.1 Random Bit Generation, or
FCS_RNG.1 Generation of random numbers]

FCS_CKM.1.1/SKG The TSF shall generate **symmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [**selection**: *cryptographic key generation algorithm*] and specified cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_CKM.1/SKG.

Table 2: Recommended choices for FCS_CKM.1/SKG

Identifier	Cryptographic Key Generation Algorithm	Cryptographic Key Sizes	List of Standards
RSK	Direct Generation from a Random Bit Generator as specified in FCS_RBG.1	[**selection**: 128, 192, 256, 512] bits	NIST SP 800-133 Revision 2 (Section 6.1).[Direct generation of symmetric keys]

3.4. FCS_CKM.2 Cryptographic Key Distribution

Catalog Guidance Notes

Key distribution (or key transport) is a key establishment scheme in which one party creates a key and sends it to another party.

Key distribution methods cover both the transmission and reception of keys. Although many products support both the transmission and reception of keys, it is not unusual to find that constrained environments only support one or the other.

FCS_CKM.2 Cryptographic Key Distribution

FCS_CKM.2	Cryptographic Key Distribution

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation, or FCS_CKM_EXT.8 Password-based key derivation, or FCS_CKM_EXT.3 Cryptographic Key Access] FCS_CKM.6 Timing and event of cryptographic key destruction [FCS_COP.1/KeyEncap Key Encapsulation, or FCS_COP.1/KeyWrap Key Wrapping, or FTP_PRO.1 Trusted Channel Protocol]

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [**selection**: key encapsulation, key wrapping, encrypted channels] that meets the following: [none].

Application Note:
If “key encapsulation” is selected, FCS_COP.1/KeyEncap must be claimed, which specifies the relevant list of standards.

If “key wrapping” is selected, FCS_COP.1/KeyWrap must be claimed, which specifies the relevant list of standards.

If “encrypted channels” is selected, FTP_PRO.1 must be claimed, which specifies the relevant list of standards.

3.5. FCS_CKM_EXT.3 Cryptographic Key Access

Catalog Guidance Notes

FCS_CKM_EXT.3 cryptographic key access applies primarily to the storage of keys for future use and retrieval of keys for immediate use by the TOE. There may be some overlap in primitives used in other SFRs, but the end goals here are to protect the confidentiality and authenticity of the keys while in storage.

This SFR recasts FCS_CKM.3.1 of CC:2022 Part 2 to place the emphasis on key access methods.

It may be necessary to combine this component with other components to ensure the confidentiality, integrity, and authenticity of the key while it is outside the control of the TOE.

FCS_CKM_EXT.3 Cryptographic Key Access

FCS_CKM_EXT.3	Cryptographic Key Access

Hierarchical to: No other components.
Dependencies: [FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.8 Password-based key derivation],
FCS_CKM.6 Timing and event of cryptographic key destruction
[FCS_COP.1/KeyEncap Key Encapsulation, or
FCS_COP.1/KeyWrap Key Wrapping, or
FCS_COP.1/SKC Symmetric Key Cryptography, or
FCS_COP.1/AEAD Authenticated Encryption with Associated Data]

FCS_CKM_EXT.3.1 The TSF shall use specified cryptographic key access methods **[selection:** key encapsulation, key wrapping, key encryption**]** to access keys when performing **[selection:** cryptographic key archival, cryptographic key backup, cryptographic key escrow, cryptographic key recovery, cryptographic key import, cryptographic key export**]**.

Application Note:
If “key encapsulation” is selected, FCS_COP.1/KeyEncap must be claimed.

If “key wrapping” is selected, FCS_COP.1/KeyWrap must be claimed.

If “key encryption” is selected, FCS_COP.1/SKC or FCS_COP.1/AEAD must be claimed.

3.6. FCS_CKM.5 Cryptographic Key Derivation

Catalog Guidance Notes

FCS_CKM.5 Cryptographic Key Derivation covers keys derived using specified cryptographic algorithms. The input to the cryptographic algorithms may be from an entropy source or from other sources. Passwords and pass phrases as input are special cases of key derivation with limited entropy input, which are addressed in FCS_CKM_EXT.8.

The output may be used as a symmetric key or for other cryptographic purposes, such as initialization vectors, authentication secrets, HMAC keys, KMAC keys, secret IVs, and secret seeds.

The protocol- and application-specific KDFs specified in NIST SP 800-135r1 (e.g., IKE, TLS) do not appear in this catalog.

FCS_CKM.5 Cryptographic Key Derivation

FCS_CKM.5	**Cryptographic Key Derivation**

Hierarchical to: No other components.

Dependencies: [FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic Operation]
FCS_CKM.6 Timing and event of cryptographic key destruction
[FCS_COP.1/CMAC Cryptographic Operation - CMAC, or
FCS_COP.1/Hash Cryptographic Operation - Hashing, or
FCS_COP.1/KeyedHash Cryptographic Operation - Keyed hash, or
FCS_COP.1/SKC Cryptographic Operation - Symmetric key cryptography, or
FCS_COP.1/AEAD Cryptographic Operation – Authenticated Encryption with Associated Data]

FCS_CKM.5.1 The TSF shall derive cryptographic keys [**selection**: *key type*] from [**selection**: *input parameters*] in accordance with a specified cryptographic key derivation algorithm [**selection**: *key derivation algorithm*] and specified cryptographic key sizes [**selection**: *key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_CKM.5.

Table 3: Recommended choices for FCS_CKM.5

Key Type	Input Parameters	Key Derivation Algorithm	Key Sizes	List of Standards
KDF-CTR	[**selection**: Direct Generation from a Random Bit Generator as specified in FCS_RBG.1, Concatenated keys]	KPF2 - KDF in Counter Mode using [**selection**: AES-128-CMAC; AES-192 -CMAC; AES-256 -CMAC; Camellia-128-CMAC; Camellia-192-CMAC; Camellia-256-CMAC; CMAC-HIGHT-128; CMAC-LEA-128; CMAC-LEA-256; CMAC-SEED-128; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF	[**selection**: 128, 192, 256, 512] bits	[**selection**: ISO/IEC 11770-6:2016 (Subclause 7.3.2) [KPF2], NIST SP 800-108 Revision 1 Update 1 (Section 4.1) [KDF in Counter Mode]]

Key Type	Input Parameters	Key Derivation Algorithm	Key Sizes	List of Standards
KDF-FB	[**selection**: Direct Generation from a Random Bit Generator as specified in FCS_RBG.1, Concatenated keys]	KPF3 - KDF in Feedback Mode using [**selection**: AES-128 -CMAC; AES-192 -CMAC; AES-256 -CMAC; Camellia-128-CMAC; Camellia-192-CMAC; Camellia-256-CMAC; CMAC-HIGHT-128; CMAC-LEA-128; CMAC-LEA-256; CMAC-SEED-128; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF	[**selection**: 128, 192, 256, 512] bits	[**selection**: ISO/IEC 11770-6:2016 (Subclause 7.3.3) [KPF3], NIST SP 800-108 Revision 1 Update 1 (Section 4.2) [KDF in Feedback Mode]]
KDF-DPI	[**selection**: Direct Generation from a Random Bit Generator as specified in FCS_RBG.1, Concatenated keys]	KPF4 - KDF in Double-Pipeline Iteration Mode using [**selection**: AES-128-CMAC; AES-192-CMAC; AES-256-CMAC, Camellia-128-CMAC; Camellia-192-CMAC; Camellia-256-CMAC; CMAC-HIGHT-128; CMAC-LEA-128; CMAC-LEA-256; CMAC-SEED-128; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as the PRF	[**selection**: 128, 192, 256, 512] bits	[**selection**: ISO/IEC 11770-6:2016 (Subclause 7.3.4) [KPF4], NIST SP 800-108 Revision 1 Update 1 (Section 4.3) [KDF in Double-Pipeline Iteration Mode]]
KDF-XOR	More than one intermediary keys	exclusive OR (XOR)	[**selection**: 128, 192, 256, 512] bits	N/A
KDF-ENC	Two keys	Encrypting using an algorithm specified in [**selection**: FCS_COP.1/SKC, FCS_COP.1/AEAD]	[**selection**: 128, 192, 256, 512] bits	N/A
KDF-HASH	Shared secret	Hash function from FCS_COP.1/Hash	[**selection**: 128, 192, 256, 512] bits	NIST SP 800-56C Revision 2 (Section 4.1, Option 1) [One-Step Key Derivation]
KDF-MAC-1S	Shared secret, salt, output length, fixed information	Keyed Hash function from FCS_COP.1/ KeyedHash	[**selection**: 128, 192, 256, 512] bits	NIST SP 800-56C Revision 2 (Section 4.1, Options 2, 3) [One-Step Key Derivation]

Key Type	Input Parameters	Key Derivation Algorithm	Key Sizes	List of Standards
KDF-MAC-2S	Shared secret, salt, IV, output length, fixed information	MAC Step [**selection**: AES-128-CMAC; AES-192-CMAC; AES-256-CMAC; Camellia-128-CMAC; Camellia-192-CMAC; Camellia-256-CMAC; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as randomness extraction and; KDF Step [**selection**: KDF-CTR, KDF-FB, KDF-DPI] using [**selection**: AES-128-CMAC; AES-192-CMAC; AES-256-CMAC; Camellia-128-CMAC; Camellia-192-CMAC; Camellia-256-CMAC; HMAC-SHA-1; HMAC-SHA-256; HMAC-SHA-512] as PRF	[**selection**: 128, 192, 256, 512] bits	NIST SP 800-56C Revision 2 (Section 5) [Two-Step Key Derivation]
KDF-KMAC	Key, context string, output length, label	[**selection**: KMAC128, KMAC256]	[**selection**: 128, 192, 256, 512] bits	NIST SP 800-108 Revision 1 Update 1 (Section 4.4 “KDF Using KMAC”)

Application Note:
In KDF-MAC-2S, if a CMAC is selected in the MAC step, then select AES-128-CMAC or Camellia-128-CMAC in the KDF step and select 128 as the output key size. If HMAC is selected in the MAC step, then select the same HMAC in the KDF.

The respective FCS_COP.1 component must be claimed for each primitive selected in *key derivation algorithm.*

3.7. FCS_CKM.6 Timing and Event of Cryptographic Key Destruction

FCS_CKM.6 Timing and Event of Cryptographic Key Destruction

FCS_CKM.6	Cryptographic Key Destruction

Hierarchical to: No other components.

Dependencies: [FDP_ITC.1 Import of user data without security attributes,
or
FDP_ITC.2 Import of user data with security attributes,
or
FCS_CKM.1 Cryptographic key generation, or
FCS_CKM_EXT.3 Cryptographic key access, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic key agreement, or
FCS_CKM_EXT.8 Password-based key derivation]

FCS_CKM.6.1 The TSF shall destroy [**assignment**: *list of cryptographic keys (including keying material)*] when [**selection**: no longer needed, [**assignment***: other circumstances for key or keying material destruction*]].

Application Note:
The TOE will have mechanisms to destroy keys, including intermediate keys and key material, by using an approved method as specified in FCS_CKM.6.2. Examples of keys include intermediate keys, leaf keys, encryption keys, and signing keys. Key material includes seeds, authentication secrets, passwords, PINs, and other secret values used to derive keys. The ST Author shall list all such keys and keying material that are subject to destruction in the first assignment.

This SFR does not apply to the public component of asymmetric key pairs or to keys that are permitted to remain stored, such as device identification keys.

FCS_CKM.6.2 The TSF shall destroy cryptographic keys and keying material specified by FCS_CKM.6.1 in accordance with a specified cryptographic key destruction method **[selection:**

1. For volatile memory, the destruction shall be executed by a [**selection:**
   a. single overwrite consisting of [**selection:**
      i. a pseudo-random pattern using the TSF's RBG,
      ii. zeroes,
      iii. ones,
      iv. a new value of a key,
      v. [**assignment:** *some value that does not contain any CSP*]],
   b. removal of power to the memory,
   c. removal of all references to the key directly followed by a request for garbage collection];
2. For non-volatile memory [**selection:**
   a. that employs a wear-leveling algorithm, the destruction shall be executed by a [**selection:**
      i. single overwrite consisting of [**selection:** zeroes, ones, pseudo-random pattern, a new value of a key of the same size, [**assignment:** *some value that does not contain any CSP*]],
      ii. block erase];
   b. that does not employ a wear-leveling algorithm, the destruction shall be executed by a [**selection:**

i. [**selection:** single, [**assignment***: ST author defined multi-pass*]] overwrite consisting of [**selection:** zeros, ones, pseudo-random pattern, a new value of a key of the same size, [**assignment:** *some value that does not contain any CSP*]] followed by a read-verify. If the read-verification of the overwritten data fails, the process shall be repeated up to [**assignment:** *number of times to attempt overwrite*] times, whereupon an error is returned.

ii. block erase]

]

] that meets the following: [*no standard*].

Application Note:
In the case of volatile memory, the selection "removal of all references to the key directly followed by a request for garbage collection" is used in a situation where the TSF cannot address the specific physical memory locations holding the data to be erased and therefore relies on addressing logical addresses (which frees the relevant physical addresses holding the old data) and then requesting the platform to ensure that the data in the physical addresses is no longer available for reading (i.e. the "garbage collection" referred to in the SFR text).

The selection for destruction of data in non-volatile memory includes block erase as an option, and this option applies only to flash memory. A block erase does not require a read verify, since the mappings of logical addresses to the erased memory locations are erased, as well as the data itself.

Some selections allow the assignment of "some value that does not contain any CSP." This means that the TOE uses some specified data not drawn from an RBG meeting FCS_RBG requirements, and not being any of the values listed as other selection options. The point of the phrase "does not contain any CSP" is to ensure that the overwritten data is carefully selected, and not taken from a general pool that might contain data that itself requires confidentiality protection.

3.8. FCS_CKM_EXT.7 Cryptographic Key Agreement

Catalog Guidance Notes

This component contains methods for multi-party key agreement in which two or more parties contribute material used to derive the shared key used by each party to encrypt and decrypt incoming and outgoing messages. TOEs can use the keys as symmetric keys, keyed-hash keys, or cryptographic keys for key derivation functions.

FCS_CKM_EXT.7 Cryptographic Key Agreement

FCS_CKM_EXT.7	**Cryptographic Key Agreement**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or

FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.8 Password-based key derivation]
[FCS_CKM.2 Cryptographic key distribution, or
FCS_COP.1 Cryptographic operation]
FCS_CKM.6 Timing and event of cryptographic key destruction
[FCS_COP.1/AEAD Authenticated encryption with associated data, or
FCS_COP.1/CMAC CMAC, or
FCS_COP.1/Hash Hashing, or
FCS_COP.1/KeyedHash, Keyed Hashing, or
FCS_COP.1/SKC Symmetric Key Cryptography, or
no other dependencies]

FCS_CKM_EXT.7.1 The TSF shall derive shared cryptographic keys with input from multiple parties in accordance with specified cryptographic key agreement algorithms [**selection**: *cryptographic algorithm*] and specified cryptographic parameters [**selection**: *cryptographic parameters*] that meets the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_CKM_EXT.7.

Table 4: Recommended choices for FCS_CKM_EXT.7

Identifier	Cryptographic Algorithm	Cryptographic Parameters	List of Standards
KAS2	RSA	Modulus Size [**selection**: 2048, 3072, 4096, 6144, 8192] bits	NIST SP 800-56B Revision 2 (Section 8.3) [KAS2]
DH	Finite Field Cryptography Diffie-Hellman	Static domain parameters approved for [**selection**: IKE groups [**selection**: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192], TLS groups [**selection**: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192]]]	NIST SP 800-56A Revision 3 (Section 5.7.1.1) [DH] [**selection**: RFC 3526 [IKE Groups], RFC 7919 [TLS Groups]]
ECDH	Elliptic Curve Diffie-Hellman	Elliptic Curve [**selection**: P-256, brainpoolP256r1, P-384, brainpoolP384r1, P-521, brainpoolP512r1]	NIST SP 800-56A Revision 3 (Section 5.7.1.2) [ECDH] [**selection**: NIST SP 800-186 (Section 3.2.1) [NIST Curves], RFC 5639 (Section 3) [Brainpool Curves]]

Identifier	Cryptographic Algorithm	Cryptographic Parameters	List of Standards
ECDH-Ed	ECDH with Montgomery Curves	Domain parameters approved for elliptic curves [**selection**: curve25519, curve448]	RFC 7748 (Section 5) [ECDH-Ed] NIST SP 800-186 (Section 3.2.2) [Montgomery Curves]

3.9. FCS_CKM_EXT.8 Password-Based Key Derivation

Catalog Guidance Notes

Password-based key derivation is different from regular key derivation in that passwords have very limited entropy. As a result, one must add additional constraints, work, or entropy to achieve acceptable levels of security when using password-based key derivation algorithms. This component only adds work through increased iterations and use of salts; it does not consider additional constraints or entropy.

This component may also be used to condition passwords in the context of password-based authentication. The output of the password-based key derivation function is not directly used as a cryptographic key, but only stored as a reference value (commonly called "password hash") to compare against when performing authentication. The "cryptographic key size" selected in this element must correspond to the length of the password hash.

See Annex B of this catalog for additional guidance regarding the security of password-based derived keys.

FCS_CKM_EXT.8 Password-Based Key Derivation

FCS_CKM_EXT.8	Password-Based Key Derivation

Hierarchical to: No other components.

Dependencies:

[FCS_CKM.2 Cryptographic key distribution, or
FCS_COP.1 Cryptographic operation or
FCS_CKM_EXT.7 Cryptographic Key Agreement]
FCS_CKM.6 Timing and event of cryptographic key destruction
FCS_OTV_EXT.1 One-Time Value Generation

FCS_CKM_EXT.8.1 The TSF shall perform password-based key derivation functions in accordance with a specified cryptographic algorithm [HMAC-[**selection***:* SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512]], with iteration count of [**assignment***: number of iterations*] using a randomly generated salt of length **[assignment**: *equal to or greater than 128*] and output cryptographic key sizes [**selection**: 128, 192, 256, 512] bits that meet the following standard: [NIST SP 800-132 (Section 5.3) [PBKDF2]].

Application Note:
NIST recommends a minimum “number of iterations” of 1000 but prefers the largest number feasible given performance constraints.

NIST recommends that the randomly generated portion of the salt have length of at least 128 bits and must be derived from a Random Bit Generation. Therefore FCS_OTV_EXT.1 must be claimed.

4. Cryptographic Operation (FCS_COP)

4.1. Catalog Guidance Notes for Family FCS_COP

SFRs under FCS_COP pertain to cryptographic operations. Such operations generally involve ensuring the authenticity or confidentiality of data. Typical cryptographic operations include encryption/decryption, digital signature generation/verification, and hashing. In this catalog, these operations are specified in eleven iterations of FCS_COP.1.

4.1.1. Data Encryption and Authentication

For data encryption without built-in authentication, include FCS_COP.1/SKC: Symmetric-Key Encryption. This SFR covers the CBC, CTR, XTS, CFB, OFB modes of symmetric-key cryptographic algorithms.

For authenticated encryption, include FCS_COP.1/AEAD: Authenticated Encryption with Associated Data. This SFR covers CCM and GCM modes of symmetric-key cryptographic algorithms. Alternatively use FCS_COP.1/SKC with FCS_COP.1/CMAC or FCS_COP.1/KeyedHash.

For authentication without encryption, include FCS_COP1/CMAC. This SFR covers the CMAC mode of symmetric-key cryptographic algorithms.

4.1.2. Key Encryption

For key encryption using asymmetric algorithms such as RSA, include FCS_COP.1/KeyEncap.

For key encryption using symmetric algorithms, include FCS_COP.1/KeyWrap. This SFR covers KW and KWP modes of symmetric cryptographic algorithms, as well as CCM and GCM modes when used for key encryption.

4.1.3. Hashing

For SHA and SHA3 hashes, include FCS_COP.1/Hash.

For Keyed Hashes, include FCS_COP.1/KeyedHash. This SFR covers HMAC and KMAC.

For extended hash output, include FCS_COP.1/XOF: Extendable-Output Functions. This SFR covers the SHAKE and KMACXOF algorithms.

4.1.4. Digital Signature Generation/Verification

For digital signature operations, include FCS_COP.1/SigGen and FCS_COP.1/SigVer.

4.2. FCS_COP.1/AEAD Cryptograpic Operation - Authenticated Encryption with Associated Data

Catalog Guidance Notes

For authenticated encryption, include FCS_COP.1/AEAD: Authenticated Encryption with Associated Data.

FCS_COP.1/AEAD Cryptographic Operation – Authenticated Encryption with Associated Data

FCS_COP.1/AEAD	**Cryptographic Operation – Authenticated Encryption with Associated Data**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic key agreement, or
FCS_CKM_EXT.8 Password-based key derivation]
FCS_CKM.6 Timing and event of cryptographic key destruction
FCS_OTV_EXT.1 One Time Value.

FCS_COP.1.1/AEAD The TSF shall perform [authenticated encryption with associated data] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/AEAD.

Table 5: Recommended choices for FCS_COP.1/AEAD

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AES-CCM	AES in CCM mode with non-repeating nonce, minimum size of 64 bits	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM]
AES-GCM	AES in GCM mode with non-repeating IVs using [**selection**: *deterministic*, *RBG-based*] IV construction; the tag must be of length [**selection:** 96, 104, 112, 120, or 128] bits.	[**selection**: 128, 192, 256], bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM]
CAM-CCM	Camellia in CCM mode with non-repeating nonce, minimum size of 64 bits	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia]

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
			[**selection**: ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM]
CAM-GCM	Camellia in GCM mode with non-repeating IVs using [**selection**: *deterministic, RBG-based*] IV construction; the tag must be of length [**selection:** 96, 104, 112, 120, or 128] bits.	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM]
SEED-CCM	SEED in CCM mode with unpredictable, non-repeating nonce, minimum size of 64 bits	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM]
SEED-GCM	SEED in GCM mode with non-repeating IVs using [**selection**: *deterministic, RBG-based*] IV construction; the tag must be of length [**selection:** 96, 104, 112, 120, or 128] bits.	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM]
LEA-CCM	LEA in CCM mode with unpredictable, non-repeating nonce, minimum size of 64 bits	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3 [LEA] [**selection**: ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM]
LEA-GCM	LEA in GCM mode with non-repeating IVs using [**selection**: *deterministic, RBG-based*] IV construction; the tag must be of length [**selection:** 96, 104, 112, 120, or 128] bits.	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3 [LEA] [**selection**: ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM]

Application Note:
If the selected cryptographic algorithm requires an IV or nonce, then FCS_OTV_EXT.1 must be claimed.

4.3. FCS_COP.1/CMAC Cryptographic Operation - CMAC

FCS_COP.1/CMAC Cryptographic Operation - CMAC

FCS_COP.1/CMAC	**Cryptographic Operation - CMAC**

Hierarchical to: No other components.

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic key agreement, or
FCS_CKM_EXT.8 Password-based key derivation]
FCS_CKM.6 Timing and event of cryptographic key destruction

FCS_COP.1.1/CMAC The TSF shall perform [CMAC] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection:** *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/CMAC.

Table 6: Recommended choices for FCS_COP.1/CMAC

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AEC-CMAC	AES using CMAC mode	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 9797-1:2011 Subclause 7.6, NIST SP 800-38B] [CMAC]
CAM-CMAC	Camillia using CMAC mode	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 Subclause 5.3 [Camellia] [**selection**: ISO/IEC 9797-1:2011 Subclause 7.6, NIST SP 800-38B] [CMAC]

4.4. FCS_COP.1/Hash Cryptographic Operation - Hashing

Catalog Guidance Notes

Since there are no keys involved with hashing, there are no cryptographic key-based dependencies necessary for this component.

FCS_COP.1/Hash Cryptographic Operation – Hashing

FCS_COP.1/Hash	Cryptographic Operation - Hashing

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_COP.1.1/Hash The TSF shall perform [cryptographic hashing] in accordance with a specified cryptographic algorithm **[selection:** SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512**]** that meets the following: **[selection:** ISO/IEC 10118-3:2018 [SHA, SHA3], FIPS PUB 180-4 [SHA], FIPS PUB 202 [SHA3]**]**.

Application Note:
The hash selection should be consistent with the overall strength of the algorithm used for signature generation. For example, the TOE should choose SHA-256 for 2048-bit RSA or ECC with P-256; SHA-384 for 3072-bit RSA, 4096-bit RSA, or ECC with P-384; and SHA-512 for ECC with P-521. The ST author selects the standard based on the algorithms selected.

SHA-1 may be used as a general hash function and for the following applications: generating and verifying hash-based message authentication codes (HMACs), key derivation functions (KDFs), and random bit/number generation. SHA-1 may also be used for verifying old digital signatures and time stamps, if this is explicitly allowed by the application domain. SHA-1 should not be used in applications in which collision resistance is needed.

4.5. FCS_COP.1/KeyedHash Cryptographic Operation - Keyed Hash

FCS_COP.1/Keyed Hash Cryptographic Operation – Keyed Hash

FCS_COP.1/KeyedHash	**Cryptographic Operation - Keyed Hash**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic key agreement, or
FCS_CKM_EXT.8 Password-based key derivation]
FCS_CKM.6 Timing and event of cryptographic key destruction
[FCS_COP.1/Hash Hashing, or
FCS_COP.1/XOF Extendable-Output Function].

FCS_COP.1.1/KeyedHash The TSF shall perform [keyed hash message authentication] in accordance with a specified cryptographic algorithm **[selection:** *keyed hash algorithm***]** and cryptographic key sizes **[selection***: cryptographic key size***]** that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/KeyedHash.

Table 7: Recommended choices for FCS_COP.1/KeyedHash

Keyed Hash Algorithm	Cryptographic Key Sizes	List of Standards
HMAC-SHA-1	[**selection**: (ISO, FIPS)160, (FIPS) 128] bits	[**selection**: ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"); FIPS PUB 198-1]
HMAC-SHA-224	[**selection**: (ISO, FIPS) 224, (FIPS) 192, 128] bits	[**selection**: ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"); FIPS PUB 198-1]
HMAC-SHA-256	[**selection**: (ISO, FIPS) 256, (FIPS) 192, 128] bits	[**selection**: ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"); FIPS PUB 198-1]
HMAC-SHA-384	[**selection**: (ISO, FIPS) 384, (FIPS) 256, 192, 128] bits	[**selection**: ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"); FIPS PUB 198-1]
HMAC-SHA-512	[**selection**: (ISO, FIPS) 512, (FIPS) 384, 256, 192, 128] bits	[**selection**: ISO/IEC 9797-2:2021 (Section 7 "MAC Algorithm 2"); FIPS PUB 198-1]
KMAC128	128 bits	[**selection**: ISO/IEC 9797-2:2021 (Section 9 "MAC Algorithm 4"); NIST SP 800-185 (Section 4 "KMAC")]
KMAC256	256 bits	[**selection**: ISO/IEC 9797-2:2021, Section 9 "MAC Algorithm 4"; NIST SP 800-185, Section 4 "KMAC"]
KMACXOF128	[**assignment**: integer 256 <= Lk < 2^{2040}	[**selection**: ISO/IEC 9797-2:2021(Section 9 "MAC Algorithm 4"); NIST SP 800-185 (Section 4 "KMAC")]
KMACXOF256	[**assignment**: integer 256 <= Lk < 2^{2040}	[**selection**: ISO/IEC 9797-2:2021(Section 9 "MAC Algorithm 4"); NIST SP 800-185 (Section 4 "KMAC")]

Application Note:
The HMAC minimum key sizes in the table are specified in ISO/IEC 9797-2:2021, which requires that the minimum key size be equal to the digest size. The FIPS standard specifies no minimum or maximum key sizes, so if FIPS PUB 198-1 is selected, larger or smaller key sizes may be used. This is indicted by the parenthesized annotations in the Cryptographic Key Sizes column.

If "KMACXOF128" or "KMACXOF256" is selected as Keyed Hash Algorithm, then FCS_COP.1/XOF must be claimed.

4.6. FCS_COP.1/KeyEncap Cryptographic Operation - Key Encapsulation

Catalog Guidance Notes

Key Encapsulation is the encryption of keys with asymmetric algorithms. For key encryption using symmetric algorithms, see FCS_COP.1/KeyWrap.

FCS_COP.1/KeyEncap Cryptographic Operation – Key Encapsulation

FCS_COP.1/KeyEncap	**Cryptographic Operation- Key Encapsulation**

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation, or FCS_CKM_EXT.7 Cryptographic key agreement, or FCS_CKM_EXT.8 Password-based key derivation] FCS_CKM.6 Timing and event of cryptographic key destruction, FSC_OTV_EXT.1 One-Time Value.

FCS_COP.1.1/KeyEncap The TSF shall perform [key encapsulation] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/KeyEncap.

Table 8: Recommended choices for FCS_COP.1/KeyEncap

Identifier	Cryptographic algorithm	Key sizes	List of Standards
KAS1	KAS1 [RSA-single party]	[**selection**: 2048, 3072, 4096, 8192] bits	NIST SP 800-56B Revision 2 (Sections 6.3 & 8.2)
KTS-OAEP	KTS-OAEP [RSA-OAEP]	[**selection**: 2048, 3072, 4096, 8192] bits	NIST SP 800-56B Revision 2 (Sections 6.3 & 9)

Application Note
NIST SP 800-57 Part 1 Revision 5 Section 5.6.2 specifies that the size of key used to protect the key being transported should be at least the security strength of the key it is protecting.

4.7. FCS_COP.1/SigGen Cryptographic Operation - Signature Generation

Catalog Guidance Notes

This component is for asymmetric cryptographic algorithms that produce cryptographic signatures. For symmetric cryptographic algorithms that produce cryptographic signatures, see FCS_COP.1/KeyHash and FCS_COP.1/CMAC.

DSA is no longer approved for digital signature generation. DSA may be used to verify signatures generated prior to the implementation date of FIPS PUB 186-5. The specifications and algorithms for DSA are no longer included in FIPS PUB 186-5. They may be found in FIPS PUB 186-4.

FCS_COP.1/SigGen Cryptographic Operation – Signature Generation

FCS_COP.1/SigGen	**Cryptographic Operation - Signature Generation**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.1 Import of user data with security attributes, or
FCS_CKM.1/AKG Asymmetric cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation]
[FCS_COP.1/Hash Hashing, or
FCS_COP.1/XOF Extendable-Output Function]
FCS_OTV_EXT.1
FCS_CKM.6 Timing and event of cryptographic key destruction.

FCS_COP.1.1/SigGen The TSF shall perform [digital signature generation] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic ~~key sizes~~ algorithm parameter [**selection**: *cryptographic algorithm parameters*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/SigGen.

Table 9: Recommended choices for FCS_COP.1/SigGen

Identifier	Cryptographic Algorithm	Cryptographic Algorithm Parameters	List of Standards
RSA-PKCS	RSASSA-PKCS1-v1_5	Modulus of size [**selection**: 2048, 3072, 4096] bits, hash or XOF [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512]	RFC 8017 (Section 8.2) [PKCS #1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PKCS1-v1_5]
RSA-PSS	RSASSA-PSS	Modulus of size [**selection**: 2048, 3072, 4096] bits, hash or XOF [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256]	RFC 8017 (Section 8.1) [PKCS#1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]

Identifier	Cryptographic Algorithm	Cryptographic Algorithm Parameters	List of Standards
ECDSA	ECDSA	Elliptic Curve [**selection**: P-256, brainpoolP256r1, P-384, brainpoolP384r1, P-521, brainpoolP512r1], per-message secret number generation [**selection**: extra random bits, rejection sampling, deterministic] and hash or XOF function using [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512, SHAKE-128, SHAKE-256]	[**selection**: ISO/IEC 14888-3:2018 (Subclause 6.6), FIPS PUB 186-5 (Sections 6.3.1, 6.4.1] [ECDSA] [**selection**: RFC 5639 (Section 3) [Brainpool Curves], NIST SP-800 186 (Section 4) [NIST Curves]]
KCDSA	KCDSA	hash function using [**selection**: SHA-224, SHA-256, SHA-384, SHA-512]	ISO/IEC 14888-3:2018 (Subclause 6.3) [KCDSA]
EC-KCDSA	EC-KCDSA	Elliptic Curve [**selection**: P-224, P-256, B-233, B-283, K-233, K-283] using hash [**selection**: SHA-224, SHA-256, SHA-384, SHA-512]	ISO/IEC 14888-3:2018 (Subclause 6.7) [EC-KCDSA] NIST SP 800-186 (Section 3) [NIST Curves]
EdDSA	Edwards-Curve Digital Signature Algorithm	Domain parameters approved for elliptic curves [**selection:** Edwards25519, Edwards448]	NIST FIPS PUB 186-5 (Section 7.6) [EdDSA] RFC 8032 [Edwards Curves]
LMS	LMS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , Winternitz parameter = [**selection**: 1, 2, 4, 8], and tree height = [**selection**: 5, 10, 15, 20, 25]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
HSS	Multitree version of LMS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , Winternitz parameter = [**selection**: 1, 2, 4, 8], tree height = [**selection**: 5, 10, 15, 20, 25], and number of levels = [**selection**: 1, 2, 3, 4, 5, 6, 7, 8]	RFC 8554 [HSS] NIST SP 800-208 [parameters]

Identifier	Cryptographic Algorithm	Cryptographic Algorithm Parameters	List of Standards
XMSS	XMSS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , tree height = [**selection**: 10, 16, 20]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]
XMSS^{MT}	Multitree version of XMSS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]](total tree height, number of levels) = [**selection**: (20, 2), (20, 4), (40, 2), (40, 4), (40, 8), (60, 3), (60, 6), (60, 12)]	RFC 8391 [XMSS^{MT}] NIST SP 800-208 [parameters]

Application Note:
The dependency on FCS_OTV_EXT.1 is needed only for signature schemes that require random bits, such as ECDSA.

4.8. FCS_COP.1/SigVer Cryptographic Operation - Signature Verification

Catalog Guidance Notes

As of the publication of FIPS PUB 186-5 on 3 February 2023, DSA is no longer approved for digital signature generation. DSA may be used to verify signatures generated prior to the implementation date of FIPS PUB 186-5. The specifications and algorithms for DSA are no longer included in FIPS PUB 186-5. They can be found in FIPS PUB 186-4.

FCS_COP.1/SigVer Cryptographic Operation – Signature Verification

FCS_COP.1/SigVer	**Cryptographic Operation - Signature Verification**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation, or
FCS_CKM.5 Cryptographic key derivation, or
No other components].
[FCS_COP.1/Hash Hashing, or
FCS_COP.1/XOF Extendable-Output Function]

FCS_COP.1.1/SigVer The TSF shall perform [digital signature verification] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic ~~key sizes~~ algorithm parameters [**selection**: *cryptographic algorithm parameters*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of **FCS_COP.1/SigVer**.

Table 10: Recommended choices for FCS_COP.1/SigVer

Identifier	Cryptographic Algorithm	Cryptographic Algorithm Parameters	List of Standards
RSA-PKCS	RSASSA-PKCS1-v1_5	Modulus of size [**selection**: 2048, 3072, 4096] bits, hash or XOF [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512]	RFC 8017 (Section 8.2) [PKCS #1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PKCS1-v1_5]
RSA-PSS	RSASSA-PSS	Modulus of size [**selection**: 2048, 3072, 4096] bits, hash or XOF [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256]	RFC 8017 (Section 8.1) [PKCS#1 v2.2] FIPS PUB 186-5 (Section 5.4) [RSASSA-PSS]
DSA	DSA	Domain parameters for (L, N) = [**selection**: (2048, 224) (2048, 256), (3072, 256)] bits	FIPS PUB 186-4 (Section 4.7) [DSA Signature Verification]
ECDSA	ECDSA	Elliptic Curve [**selection**: P-256, brainpoolP256r1, P-384, brainpoolP384r1, P-521, brainpoolP512r1] using hash or XOF [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256]	[**selection**: ISO/IEC 14888-3:2018 (Subclause 6.6), FIPS PUB 186-5 (Section 6.4.2)] [ECDSA] [**selection**: RFC 5639 (Section 3) [Brainpool Curves], NIST SP 800-186 (Section 3) [NIST Curves]]
KCDSA	KCDSA	hash function using [**selection**: SHA-224, SHA-256, SHA-384, SHA-512]	ISO/IEC 14888-3:2018 (Subclause 6.3) [KCDSA]
EC-KCDSA	EC-KCDSA	Elliptic Curve [**selection**: P-224, P-256, B-233, B-283, K-233, K-283] using hash [**selection**: SHA-224, SHA-256, SHA-384, SHA-512]	ISO/IEC 14888-3:2018 (Subclause 6.7) [EC-KCDSA] NIST SP 800-186 (Section 3) [NIST Curves]
EdDSA	Edwards-Curve Digital Signature Algorithm	Domain parameters approved for elliptic curves [**selection:** Edwards25519, Edwards448]	NIST FIPS PUB 186-5 (Section 7.7) [EdDSA] RFC 8032 [Edwards Curves]

Identifier	Cryptographic Algorithm	Cryptographic Algorithm Parameters	List of Standards
LMS	LMS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , Winternitz parameter = [**selection**: 1, 2, 4, 8], and tree height = [**selection**: 5, 10, 15, 20, 25]	RFC 8554 [LMS] NIST SP 800-208 [parameters]
HSS	Multitree version of LMS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , Winternitz parameter = [**selection**: 1, 2, 4, 8], tree height = [**selection**: 5, 10, 15, 20, 25], and number of levels = [**selection**: 1, 2, 3, 4, 5, 6, 7, 8]	RFC 8554 [HSS] NIST SP 800-208 [parameters]
XMSS	XMSS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]] , tree height = [**selection**: 10, 16, 20]	RFC 8391 [XMSS] NIST SP 800-208 [parameters]
XMSS^{MT}	Multitree version of XMSS	Private key size = [**selection**: 192 bits with [**selection**: SHA-256/192, SHAKE256/192], 256 bits with [**selection**: SHA-256, SHAKE256]](total tree height, number of levels) = [**selection**: (20, 2), (20, 4), (40, 2), (40, 4), (40, 8), (60, 3), (60, 6), (60, 12)]	RFC 8391 [XMSS^{MT}] NIST SP 800-208 [parameters]

Application Note:
The TOE may contain a public key which is integrity protected (e.g., in hardware), in which case the FDP_ITC.1 and FDP_ITC.2 dependencies do not apply. In this case, no dependencies may be chosen. For signature verifications, private keys are not necessary, so there are no dependencies required for generating or destroying cryptographic keys.

4.9. FCS_COP.1/KeyWrap Cryptographic Operation - Key Wrapping

Catalog Guidance Notes

Key Wrapping is the encryption of keys with symmetric algorithms.

FCS_COP.1/KeyWrap Cryptographic Operation - Key Wrapping

FCS_COP.1/KeyWrap	**Cryptographic Operation - Key Wrapping**

Hierarchical to: No other components.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation, or

FCS_CKM.5 Cryptographic key derivation, or
FCS_CKM_EXT.7 Cryptographic key agreement, or
FCS_CKM_EXT.8 Password-based key derivation]
FCS_CKM.6 Timing and event of cryptographic key destruction
FCS_COP.1/SKC Symmetric key cryptography.

FCS_COP.1.1/KeyWrap The TSF shall perform [key wrapping] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1.1/KeyWrap.

Table 11: Recommended choices for FCS_COP.1/KeyWrapw

Identifier	Cryptographic algorithm	Cryptographic key sizes	List of standards
KW	[**selection**: AES, CAM, SEED, LEA] in KW mode	[**selection**: (AES, CAM, SEED, LEA) 128, (AES, CAM, LEA) 192, (AES, CAM, LEA) 256] bits	[**selection:** ISO/IEC 19772:2020 (clause 6), NIST SP 800-38F (Section 6.2)] [KW mode]
KWP	[**selection**: AES, CAM, SEED, LEA] in KWP mode	[**selection**: (AES, CAM, SEED, LEA) 128, (AES, CAM, LEA) 192, (AES, CAM, LEA) 256] bits	NIST SP 800-38F (Section 6.3) [KWP mode]
CCM	[**selection:** AES, CAM, LEA, SEED] in CCM mode with non-repeating nonce, minimum size of 64 bits	[**selection**: (AES, CAM, SEED, LEA) 128, (AES, CAM, LEA) 192, (AES, CAM, LEA) 256] bits	[**selection**: ISO/IEC 19772:2020 (Clause 7), NIST SP 800-38C] [CCM mode]
GCM	[**selection:** AES, CAM, LEA, SEED] in GCM mode with non-repeating IVs IV length must be equal to 96 bits; the deterministic IV construction method [SP800-38D, Section 8.2.1] must be used; the MAC length t must be one of the values 96, 104, 112, 120, and 128 bits.	[**selection**: (AES, CAM, SEED, LEA) 128, (AES, CAM, LEA) 192, (AES, CAM, LEA) 256] bits	[**selection**: ISO/IEC 19772:2020 (Clause 10), NIST SP 800-38D] [GCM mode]

Application Note
NIST 800-57p1rev5 sec. 5.6.2 specifies that the size of key used to protect the key being transported should be at least the security strength of the key it is protecting.

The SEED algorithm supports keys of size 128 bits only.

4.10. FCS_COP.1/SKC Cryptographic Operation - Symmetric-Key Cryptography

Catalog Guidance Notes

The modes covered in FCS_COP.1/SKC are used for symmetric-key cryptography without authentication.

FCS_COP.1/SKC Cryptographic Operation – Symmetric-Key Cryptography

FCS_COP.1/SKC	**Cryptographic Operation – Symmetric-Key Cryptography**

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation, or FCS_CKM_EXT.7 Cryptographic key agreement, or FCS_CKM_EXT.8 Password-based key derivation] FCS_CKM.6 Timing and event of cryptographic key destruction FCS_OTV_EXT.1 One Time Value.

FCS_COP.1.1/SKC The TSF shall perform [symmetric-key encryption/decryption] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and cryptographic key sizes [**selection**: *cryptographic key sizes*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/SKC.

Table 12: Recommended choices for FCS_COP.1/SKC

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
AES-CBC	AES in CBC mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
XTS-AES	AES in XTS mode with unique tweak values that are consecutive non-negative integers starting at an arbitrary non-negative integer	[**selection**: 256, 512] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: IEEE Std. 1619-2018, NIST SP 800-38E] [XTS]
AES-CTR	AES in Counter Mode with a non-repeating initial counter and with no repeated use of counter values across multiple messages with the same secret key.	[**selection**: 128, 192, 256] bits	[**selection**: ISO/IEC 18033-3:2010 (Subclause 5.2), FIPS PUB 197] [AES] [**selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]
CAM-CBC	Camellia in CBC mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
CAM-CFB	Camellia in CFB mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 10116:2017 (Clause 8), NIST SP 800-38A] [CFB]
CAM-OFB	Camellia in OFB mode with unique IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 10116:2017 (Clause 9), NIST SP 800-38A] [OFB]
XTS-CAM	Camellia in XTS mode with unique tweak values that are consecutive non-negative integers starting at an arbitrary non-negative integer	[**selection**: 256, 512] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: IEEE Std. 1619-2018, NIST SP 800-38E] [XTS]

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
CAM-CTR	Camellia in CTR mode with a non-repeating initial counter and with no repeated use of counter values across multiple messages with the same secret key.	[**selection**: 128, 192, 256] bits	ISO/IEC 18033-3:2010 (Subclause 5.3) [Camellia] [**selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]
SEED-CBC	SEED in CBC mode with non-repeating and unpredictable IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
SEED-CFB	SEED in CFB mode with non-repeating and unpredictable IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 10116:2017 (Clause 8), NIST SP 800-38A] [CFB]
SEED-OFB	SEED in OFB mode with unique IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 10116:2017 (Clause 9), NIST SP 800-38A] [OFB]
SEED-CTR	SEED in CTR mode with unique, incremental counter	128 bits	ISO/IEC 18033-3:2010 (Subclause 5.4) [SEED] [**selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]
HIGHT-CBC	HIGHT in CBC mode with non-repeating and unpredictable IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 4.5) [HIGHT] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
HIGHT-CFB	HIGHT in CFB mode with non-repeating and unpredictable IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 4.5) [HIGHT] [**selection**: ISO/IEC 10116:2017 (Clause 8), NIST SP 800-38A] [CFB]

Identifier	Cryptographic Algorithm	Cryptographic Key Sizes	List of Standards
HIGHT-OFB	HIGHT in OFB mode with unique IVs	128 bits	ISO/IEC 18033-3:2010 (Subclause 4.5) [HIGHT] [**selection**: ISO/IEC 10116:2017 (Clause 9), NIST SP 800-38A] [OFB]
HIGHT-CTR	HIGHT in CTR mode with unique, incremental counter	128 bits	ISO/IEC 18033-3:2010 (Subclause 4.5) [HIGHT] [**selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]
LEA-CBC	LEA in CBC mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3) [LEA] [**selection**: ISO/IEC 10116:2017 (Clause 7), NIST SP 800-38A] [CBC]
LEA-CFB	LEA in CFB mode with non-repeating and unpredictable IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3) [LEA] [**selection**: ISO/IEC 10116:2017 (Clause 8), NIST SP 800-38A] [CFB]
LEA-OFB	LEA in OFB mode with unique IVs	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3) [LEA] [**selection**: ISO/IEC 10116:2017 (Clause 9), NIST SP 800-38A] [OFB]
LEA-CTR	LEA in CTR mode with unique, incremental counter	[**selection**: 128, 192, 256] bits	ISO/IEC 29192-2:2019 (Subclause 6.3) [LEA] [**selection**: ISO/IEC 10116:2017 (Clause 10), NIST SP 800-38A] [CTR]

Application Note:
If the selected “cryptographic algorithm" requires an IV, counter, or tweak value, then FCS_OTV_EXT.1 must be claimed.

4.11. FCS_COP.1/XOF Extendable-Output Function

FCS_COP.1/XOF Extendable-Output Function

FCS_COP.1/XOF	Cryptographic Operations (Extendable-Output Function)

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation].

FCS_COP.1/XOF The TSF shall perform [extendable-output function] in accordance with a specified cryptographic algorithm [**selection**: *cryptographic algorithm*] and *parameters* [**selection**: *parameters*] that meet the following: [**selection**: *list of standards*].

The following table provides the recommended choices for completion of the selection operations of FCS_COP.1/XOF.

Table 13: Recommended choices for FCS_COP.1/XOF

Cryptographic algorithm	Parameters	List of standards
cSHAKE	Output length *d* = [**selection**: 128, 256] bits and function [**selection**: SHAKE*d*, KECCAK[*2d*]]	NIST SP 800-185 Section 3 [cSHAKE], Section 6.2 [SHAKE] NIST FIPS PUB 202 Section 5 [KECCAK]
KMACXOF	Output length *d* = [**selection**: 128, 256] bits	NIST SP 800-185 Section 4.3.1 [KMACXOF]
SHAKE	Output length *d* = [**selection**: 128, 256] bits	NIST FIPS PUB 202 Section 6.2 [SHAKE]

Application Note:
The functions in cSHAKE depend on the output length *d*. i.e. SHAKE*d* is either SHAKE128 for $d = 128$ or SHAKE256 for $d = 256$. Similarly, KECCAK[*2d*] is either KECCAK[256] for $d = 128$ or KECCAK[512] for $d = 256$. Note that KECCAK is a cryptographic primitive which should have no direct interface exposed to the user of the TOE.

5. One-Time Value Generation (FCS_OTV)

5.1. Catalog Guidance Notes for Family FCS_OTV

The lone SFR under FCS_OTV pertains to generation or derivation of one-time use values, such as initialization vectors, nonces, tweak values, and salts.

5.2. FCS_OTV_EXT.1 One-Time Value

Catalog Guidance Notes

TSFs frequently generate cryptographic one-time values, often non-secret, such as nonces, IVs, salts, and initial counters (sometimes called initial sequential nonces) using the output of an RBG specified in FCS_RBG.1. If the TSF is generating OTVs, then this SFR is used.

Salts help protect against dictionary and other precomputation attacks. Systems often prepend or append salts to passwords and other long-term, potentially guessable values to increase the size of a dictionary an attacker must build to attack it. Salts, once associated with a password, generally do not change for the life of that password. Salts should also be unique for each password and should not be reused. Therefore, systems should randomly generate salts with sufficient size such that the combined entropy of both the salt and the password meets the minimal key strength sizes of the chosen algorithms.

Nonces help protect against replay attacks in cryptographic authentication protocols and some encryption modes. A nonce should never repeat. Using a sequence of nonces with a counter embedded in the value will ensure a nonce will never repeat. In protocol sessions that require multiple nonces, using sequential nonces that increment for each message—the receiver can check for and accept only an increase in the nonce value to verify that the message has not been replayed. In some protocols, the initial sequential nonce needs only to be sent once at the beginning of the session and the receiver can predict the remaining nonces in that session, which saves transmission bandwidth. Randomly generated nonces protect against attacks against sessions in which multiple keys are expected to be used. Therefore, nonces should be both randomly generated and never repeat. However, sequential nonces may be predictable. NIST provides additional guidance for the composition of a nonce in NIST SP 800-38c, NIST SP 800-56A Revision 3, NIST SP 800-56B Revision 2, NIST SP 800-63B, and NIST SP 800-90A Revision 1.

Initialization Vectors (IVs) help protect against attacks which depend on the reuse of static keys. Certain encryption modes often require IVs. They should be randomly generated in a nonpredictable way, cannot be sequential, and cannot repeat.

Each algorithm and mode have varying guidance on the lengths of the salts, nonces, and initialization vectors used therein. Please consult the referenced standards documents for the appropriate guidance for each.

FCS_OTV_EXT.1 One-Time Value

FCS_OTV_EXT.1	One-Time Value

Hierarchical to: No other components.
Dependencies: FCS_RBG.1 Random Bit Generators
[FCS_COP.1/HMAC Key Hash, or
FCS_COP.1/SKC Symmetric key cryptography, or
FCS_CKM.5 Key Derivation, or
FCS_CKM_EXT.8 Password-Based Key Derivation, or
FCS_COP.1/CMAC CMAC, or
FCS_COP.1/KeyWrap Key Wrapping
FCS_COP.1/AEAD Authenticated Encryption with Associated Data, or
FCS_COP.1/KeyEncap Key Encapsulation]

FCS_OTV_EXT.1.1 The TSF shall perform *cryptographic one-time value generation* for [**selection**: *algorithm or mode*] using the output of a [**selection:** *random bit generator as defined in FCS_RBG.1, deterministic OTV construction*, [**assignment:** *OTV construction method*]] and sizes of length that meet the following: [**selection**: *list of standards*]

The following table provides the recommended choices for completion of the selection operations of FCS_OTV_EXT.1.

Table 14: Recommended choices and guidance for FCS_OTV_EXT.1

Algorithm or Mode	List of Standards	Notes
HMAC	FIPS PUB 198-1, NIST SP 800-56C Revision 2	Depending on the use case, salts can be secret or known, randomly generated or all zero. Secret IVs may be required, e.g., for key derivation. Refer to the relevant standards for your use case.
KMAC	NIST SP 800-185, NIST SP 800-56C Revision 2	Depending on the use case, salts can be secret or known, randomly generated or all zero. Secret IVs may be required, e.g., for key derivation. Refer to the relevant standards for your use case.
KDF	NIST SP 800-108 Revision 1, NIST SP 800-135 Revision 1, ISO/IEC 11770-6:2016 (Subclause 7.3.2)	Salts and IVs are generated as directed for HMAC, AES, and CAM cryptographic algorithms. Refer to the relevant standards.
PBKDF	NIST SP 800-132	Salts are generated and used as directed in PBKDFs.
CTR	NIST SP 800-38A	"Initial Counter" (nonce) shall be non-repeating. No counter value shall be repeated across multiple messages with the same secret key.
CBC	NIST SP 800-38A Appendix C	Depending on the use case, IVs shall be unpredictable. Repeating IVs leak information about whether the first one or more blocks are shared between two messages, so IVs should be non-repeating in such situations. Refer to the relevant standards for your use case.

Algorithm or Mode	List of Standards	Notes
OFB	NIST SP 800-38A	IVs shall be non-repeating and shall not be generated by invoking the cipher on another IV. OFB may require the IV to be a nonce.
CFB	NIST SP 800-38A	IVs should be non-repeating as repeating IVs leak information about the first plaintext block and about common shared prefixes in messages.
XTS	NIST SP 800-38E, IEEE Std 1619-2018	Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer (i.e., sequential nonces).
CMAC	NIST SP 800-38B	IV is all zeroes.
KW, KWP	NIST SP 800-38F	Depending on the use case, nonces may be required. Please reference the relevant standards for your use case.
CCM	NIST SP 800-38C	Nonces shall be non-repeating.
GCM	NIST SP 800-38D	For RBG-based IV construction (section 8.2.2) the number of invocations of GCM shall not exceed 2^32 for a given secret key.
RSA-OAEP	NIST SP 800-56B Revision 2	Mask for padding shall be randomly generated.

Application Note:

See the algorithm- or mode-specific Notes above for guidance on completing the second selection.

6. Random Bit Generation (FCS_RBG)

6.1. Catalog Guidance Notes for Family FCS_RBG

The SFRs in FCS_RBG apply only to deterministic random bit generators and not to non-deterministic RBGs. Health tests for the RBG are specified in FPT_TST.1. In the context of these FCS_RBG SFRs, the term noise source refers to both raw noise sources as well as conditioned entropy sources, both of which must meet min-entropy requirements for initializing DRBGs.

In the context of these FCS_RBG SFRs, the term *seed* is used to mean the collection of all parameters used to initialize the DRBG. The term *seeding* has multiple meanings depending on the context. For *external seeding* and *internal seeding*, we mean external entropy source and internal entropy source. Otherwise, the term *seeding* means the process of initialization, which is distinct from reseeding.

The following components are based on the FCS_RBG family of the CC:2022 Revision 1 Part 2 with proposed corrections and interpretations from the errata CCMB-2024-07-002 Version 1.1.

6.2. FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.1	**Random Bit Generation**

Hierarchical to:	No other components.
Dependencies:	[FCS_RBG.2 Random Bit Generation (External Seeding), or FCS_RBG.3 Random Bit Generation (Internal Seeding Single Source)] FCS_COP.1/Hash Hashing FCS_COP.1/SKC Symmetric Key Cryptography FPT_FLS.1 Failure with preservation of secure state. FPT_TST.1 TSF testing

FCS_RBG.1.1 The TSF shall perform deterministic random bit generation services using [**selection**: *DRBG algorithm*] in accordance with [**selection**: *list of standards*] after initialization.

The following table provides the recommended choices for completion of the selection operations of FCS_RBG.1.

Table 15: Recommended choices for FCS_RBG.1.1

Identifier	RBG Algorithm	List of Standards
HASH_DRBG	Hash_DRBG with [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512]	[**selection:** ISO/IEC 18031: 2011 (Section C.2.2), NIST SP 800-90A Revision 1 Section 10.1.1]
HMAC_DRBG	HMAC_DRBG with [**selection**: SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512]	[**selection:** ISO/IEC 18031: 2011 (Section C.2.3), NIST SP800-90A Revision 1 Section 10.1.2]
CTR_DRBG	CTR_DRBG with [**selection:** AES-128, AES-192, AES-256, CAM-128, CAM-192, CAM-256, SEED-128, HIGHT-128, LEA-128, LEA-192, LEA-256]	[**selection:** ISO/IEC 18031: 2011 (Section C.3.2), NIST SP800-90A Revision 1 Section 10.2.1]

FCS_RBG.1.2 The TSF shall use a [**selection**: TSF entropy source [**assignment***: name of entropy source*], TSF interface for obtaining entropy] for initialization and reseeding.

FCS_RBG.1.3 The TSF shall update the DRBG state by **[selection:** reseeding, uninstantiating and re-instantiating**]** using a **[selection:** TSF entropy source **[assignment:** *name of entropy source***]**, TSF interface for obtaining entropy [**assignment**: *name of the interface*]**]** in the following situations: **[selection:**

- never,
- on demand,
- on the condition: [**assignment**: *condition*],
- after [**assignment**: *time*]]

in accordance with [**assignment**: *list of standards*].

Application Note:
No rationale is acceptable for not satisfying one of these dependencies.

If a reseeding is selected in the first selection and something other than “never” is selected in the third selection of FCS_RBG.1.3, but reseeding is not feasible, the TSF will uninstantiate RBGs, rather than produce output that is of insufficient quality. The listed standards should specify the reseed interval and procedure for uninstantiating and reseeding. The remaining selection allows the PP Author to require application-specific conditions for reseeding.

“Uninstantiate” means that the internal state of the DRBG is no longer available for use.
In the second selection of FCS_RBG.1.3, “on demand” means that a TOE presents an interface to reseed as a TSFI (e.g., an API call). The interface causes the DRBG to reseed at the request of an authorized user, either with an internal source, an external source, or from input provided through the TSFI (e.g., the API call).

6.3. FCS_RBG.2 Random Bit Generation (External Seeding)

FCS_RBG.2 Random Bit Generation (External Seeding)

FCS_RBG.2	Random Bit Generation (External Seeding)

Hierarchical to: No other components.
Dependencies: FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.2.1 The TSF shall be able to accept a minimum input of [***assignment***: *minimum input length greater than zero*] from a TSF interface for the purpose of obtaining entropy.

Application Note:
In order to maintain compliance with NIST SP 800-90A Revision 1, the TSF accepts enough bits of input from an external noise source to satisfy the entropy requirements of the DRBG. The TSF should also protect the integrity and confidentiality of the entropy it receives from the external noise source.
The TSF interface for the purpose of seeding here is the interface used to gather entropy for initializing the seed.

6.4. FCS_RBG.3 Random Bit Generation (Internal Seeding - Single Source)

FCS_RBG.3 Random Bit Generation (Internal Seeding - Single Source)

FCS_RBG.3	Random Bit Generation (Internal Seeding – Single Source)

Hierarchical to: No other components.
Dependencies: FCS_RBG.1 Random Bit Generation (RBG)
FCS_RBG.5 Random Bit Generation (Combining Noise Sources)

FCS_RBG.3.1 The TSF shall be able to seed the DRBG using a [**selection**, **choose one of**: TSF software-based entropy source, TSF hardware-based entropy source] [**assignment**: *name of entropy source*] with [**assignment**: *number of bits*] bits of min-entropy.

Application Note:
If an ST Author wishes to use multiple internal noise sources, they iterate this requirement for each noise source used by the TSF.

Hardware-based noise sources are entropy sources whose primary function is noise generation, such as ring oscillators, diodes, and thermal noise. While a TOE may use software to collect the noise from these hardware sources, these are not software-based. Software-based noise sources are those sources that have some other primary function, and the noise is a byproduct of their

normal operation. Examples of software-based noise sources are user or system-based events, reading the least significant bits from an event timer, etc.

Hardware-based noise sources may be stochastically modelled, in which case the amount of entropy is well understood. Software-based noise sources are usually less well understood and therefore will typically take a more conservative approach, gathering larger numbers of bits than required, then performing a compression function to derive the final output. Software-based noise sources often rely on an entropy estimator.

6.5. FCS_RBG.4 Random Bit Generation (Internal Seeding - Multiple Sources)

FCS_RBG.4 Random Bit Generation (Internal Seeding - Multiple Sources)

FCS_RBG.4	Random Bit Generation (Internal Seeding – Multiple Sources)

Hierarchical to:	No other components.
Dependencies:	FCS_RBG.1 Random Bit Generation (RBG) FCS_RBG.5 Random Bit Generation (Combining Entropy Sources)

FCS_RBG.4.1 The TSF shall be able to seed the DRBG using [**selection**: [**assignment***: number*] TSF software-based entropy source(s), [**assignment**: *number*] TSF hardware-based entropy source(s)].

6.6. FCS_RBG.5 Random Bit Generation (Combining Entropy Sources)

FCS_RBG.5 Random Bit Generation (Combining Entropy Sources)

FCS_RBG.5	Random Bit Generation (Combining Entropy Sources)

Hierarchical to:	No other components.
Dependencies:	FCS_RBG.1 Random Bit Generation (RBG) [FCS_RBG.2 Random Bit Generation (External Seeding), or FCS_RBG.3 Random Bit Generation (Internal Seeding - Single Source), or FCS_RBG.4 Random Bit Generation (Internal Seeding - Multiple Sources)]

FCS_RBG.5.1 The TSF shall [**selection**: hash, concatenate and hash, XOR, input into a linear feedback shift register, [**assignment**: *combining operation*]] [**selection**: output from TSF entropy source(s), input from TSF interface(s) for obtaining entropy] resulting in a minimum of

[**assignment**: *number of bits*] bits of min-entropy to create the entropy input into the derivation function as defined in [**selection**: ISO/IEC 18031: 2011, NIST SP 800-90A Revision 1]

Application Note:
One can apply NIST SP 800-90B (or AIS-31) statistical tests against internal noise sources (a.k.a. raw entropy) to confirm the min-entropy of the noise sources either in aggregate or individually. One should not apply NIST SP 800-90B (or AIS-31) statistical tests against external noise sources since the TOE is unable to enforce entropy requirements or conditioning requirements against external sources of entropy. However, the TSS may include estimates for min-entropy from external sources that contribute to the overall entropy requirements for either the DRBG or for FCS_OTV_EXT.1.

FCS_RBG.5 specifies the combining operation such that the combined min-entropy of all the internal sources and the estimated entropy of the external sources is greater than or equal to the desired entropy of the output of the combining operation. The output could be used as a nonce or a seed for a DRBG. The combining operation should avoid crushing the entropy of the sources such that the desired entropy of the output cannot be met.

The TSF interface(s) for seeding here is the interface used to gather entropy for initializing the seed.

6.7. FCS_RBG.6 Random Bit Generation Service

FCS_RBG.6 Random Bit Generation Service

FCS_RBG.6	**Random Bit Generation Service**

Hierarchical to: No other components.
Dependencies: FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.6.1 The TSF shall provide a [**selection**: hardware, software, [**assignment**: *other interface type*]] interface to make the DRBG output, as specified in FCS_RBG.1 Random Bit Generation (RBG), available as a service to entities outside of the TOE.

Annex A: Extended Component Definitions

A.1. Class FCS: Cryptographic Support

Class Description

The TSF may employ cryptographic functionality to help satisfy several high-level security objectives. These include, but are not limited to: identification and authentication, nonrepudiation, trusted path, trusted channel, and data separation. This class is used when the TOE implements cryptographic functions, the implementation of which can be in hardware, firmware and/or software.

The FCS: Cryptographic support class is composed of five families.

- FCS_CKM: Cryptographic support
- FCS_COP: Cryptographic operation
- FCS_OTV: One-time value generation
- FCS_RBG: Random bit generation
- FCS_RNG: Random number generation

A.2. Cryptographic key management (FCS_CKM)

Family Behavior

Cryptographic keys must be managed throughout their life cycle. This family is intended to support that lifecycle and consequently defines requirements for the following activities:

— cryptographic key generation;
— cryptographic key distribution;
— cryptographic key access;
— cryptographic key derivation;
— timing and event of cryptographic key destruction;
— cryptographic key agreement;
— password-based key derivation.

This family should be included whenever there are functional requirements for the management of cryptographic keys.

Component leveling and description

FCS_CKM.1 Cryptographic key generation, requires cryptographic keys to be generated in accordance with a specified algorithm and key sizes which can be based on an assigned standard.

FCS_CKM.2 Cryptographic key distribution, requires cryptographic keys to be distributed in accordance with a specified distribution method which can be based on an assigned standard.

FCS_CKM_EXT.3 Cryptographic key access, requires access to cryptographic keys stored outside the TOE to be performed in accordance with a specified access method.

FCS_CKM.5 Cryptographic key derivation, requires that the methods, standards, and parameters for key-derivation are specified.

FCS_CKM.6 Timing and event of cryptographic key destruction, requires cryptographic keys to be destroyed in accordance with specified destruction methods which can be based on an assigned standard.

FCS_CKM_EXT.7 Cryptographic key agreement, requires cryptographic keys to be derived and shared between multiple parties in accordance with a specified multi-party key derivation method which can be based on an assigned standard.

FCS_CKM_EXT.8 Password-based cryptographic key derivation, requires cryptographic keys to be derived from low-entropy password input using specified cryptographic primitives which can be based on an assigned standard.

Management of FCS_CKM_EXT.3, FCS_CKM_EXT.7, FCS_CKM_EXT.8

The following actions can be considered for the management functions in FMT:

a) there are no management activities foreseen.

Audit of FCS_CKM_EXT.3, FCS_CKM_EXT.7, FCS_CKM_EXT.8

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP, PP-Module, functional package or ST:

a) minimal: Success and failure of the activity;
b) basic: The object attribute(s), and object value(s) excluding any sensitive information.

A.2.1. FCS_CKM_EXT.3 Cryptographic key access

Component Relationships

Hierarchical to:	No other components.
Dependencies:	[FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation, or FCS_CKM_EXT.8 Password-based key derivation], FCS_CKM.6 Timing and event of cryptographic key destruction [FCS_COP.1/KeyEncap Key Encapsulation, or FCS_COP.1/KeyWrap Key Wrapping, or FCS_COP.1/SKC Symmetric Key Cryptography, or FCS_COP.1/AEAD Authenticated Encryption with Associated Data]

FCS_CKM_EXT.3.1
The TSF shall use specified cryptographic key access methods **[selection:** key encapsulation, key wrapping, key encryption**]** to access keys when performing **[selection:** cryptographic key archival,

cryptographic key backup, cryptographic key escrow, cryptographic key recovery, cryptographic key import, cryptographic key export**]**.

A.2.2. FCS_CKM_EXT.7 Cryptographic key agreement

Component Relationships

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation, or FCS_CKM.5 Cryptographic key derivation, or FCS_CKM_EXT.8 Password-based key derivation] [FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic operation] FCS_CKM.6 Timing and event of cryptographic key destruction [FCS_COP.1/AEAD Authenticated encryption with associated data, or FCS_COP.1/CMAC CMAC, or FCS_COP.1/Hash Hashing, or FCS_COP.1/KeyedHash, Keyed Hashing, or FCS_COP.1/SKC Symmetric Key Cryptography, or no other dependencies]

FCS_CKM_EXT.7.1

The TSF shall derive shared cryptographic keys with input from multiple parties in accordance with specified cryptographic key agreement algorithms [**selection**: *cryptographic algorithm*] and specified cryptographic parameters [**selection**: *cryptographic parameters*] that meets the following: [**selection**: *list of standards*].

A.2.3. FCS_CKM_EXT.8 Password-based key derivation

Component relationships

Hierarchical to:	No other components.
Dependencies:	[FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic operation or FCS_CKM_EXT.7 Cryptographic Key Agreement] FCS_CKM.6 Timing and event of cryptographic key destruction FCS_OTV_EXT.1 One-Time Value Generation

FCS_CKM_EXT.8.1

The TSF shall perform password-based key derivation functions in accordance with a specified cryptographic algorithm [*HMAC*-[**selection***:* SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-

384, SHA3-512]], with iteration count of [**assignment***: number of iterations*] using a randomly generated salt of length **[assignment**: *equal to or greater than 128*] and output cryptographic key sizes [**selection**: 128, 192, 256, 512] bits that meet the following standard: [*NIST SP 800-132 Section 5.3 (PBKDF2)*].

A.3. One-Time value generation (FCS_OTV)

Family Behavior

Cryptographic operations often require one-time values such as nonces, IVs, salts, and initial counters. These values are often non-secret.

Component leveling and description

FCS_OTV_ENT.1 One-time value generation, requires that values such as salts, nonces, IVs, and initial counters be generated using random bit generation.

Management of FCS_OTV_EXT.1

The following actions can be considered for the management functions in FMT:

a) there are no management activities foreseen.

Audit of FCS_OTV_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP, PP-Module, functional package or ST:

a) minimal: Success and failure of the activity;
b) basic: The object attribute(s), and object value(s) excluding any sensitive information.

A.3.1. FCS_OTV_EXT.1 One-time value generation

Component relationships

Hierarchical to:	No other components.
Dependencies:	FCS_RBG.1 Random Bit Generators [FCS_COP.1/HMAC Key Hash, or FCS_COP.1/SKC Symmetric key cryptography, or FCS_CKM.5 Key Derivation, or FCS_CKM_EXT.8 Password-Based Key Derivation, or FCS_COP.1/CMAC CMAC, or FCS_COP.1/KeyWrap Key Wrapping FCS_COP.1/AEAD Authenticated Encryption with Associated Data, or FCS_COP.1/KeyEncap Key Encapsulation]

FCS_OTV_EXT.1.1
The TSF shall perform *cryptographic one-time value generation* for [**selection**: *algorithm or mode*] using the output of a [**selection:** *random bit generator as defined in FCS_RBG.1, deterministic OTV construction*, [**assignment:** *OTV construction method*]] and sizes of length that meet the following: [**selection**: *list of standards*].

Annex B: Additional Guidance for Password-Based Key Derivation

FCS_CKM_EXT.8 Password-Based Key Derivation provides only for an increased number of iterations as a means of adding difficulty to exhaustion attacks against a password.

NIST recommends setting the number of iterations to some value that increases the cost for attackers but is not too inconvenient for legitimate users. The 10 million iterations suggested by NIST may seem excessive, but it takes only 1 second to process on a modern ARMv8-based device, such as a mobile phone. If an attack can be conducted off-line using the fastest available processors, it might not take even that long.

One way to mitigate against password exhaustion attacks is to combine FCS_CKM_EXT.8 with interface-based mitigations, such as those in FIA_AFL.1 Authentication Failure Handling.

Passwords can be bound to the TOE using a randomly generated secret salt that is securely stored within the TOE. This requires that an attacker guess both the secret salt and the password, which effectively thwarts off-line attacks and forces an attacker to use the TOE interface in order to attempt to guess the password.

Once an attacker is forced to use the TOE interface, password guessing can be throttled using methods specified in a requirement such as FIA_AFL.1. Perhaps by imposing time penalties for authentication failures, limiting the number of authentication attempts, or limiting the frequency of authentication attempts.

Using additional methods such as these reduces the importance of the number of iterations, and allows smaller values to be used for that parameter.

References

[1] _, *ANSI X9.63-2011 (R2017): Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography*, American National Standards Institute, 16 February 2017.

[2] _, *Common Criteria for Information Security Technology Security Evaluation, Part 1: Introduction and General Model, CC:2022, Revision 1, CCMB-2022-11-001*, Common Criteria Maintenance Board, November 2022.

[3] _, Common Criteria Maintenance Board, *Common Criteria for Information Security Technology Security Evaluation, Part 2: Security Functional Components, CC:2022, Revision 1, CCMB-2022-11-002*, Common Criteria Recognition Arrangement, November 2022.

[4] _, *Federal Information Processing Standards Publication 180-4 Secure Hash Standard (SHS)*, U.S. National Institute of Standards and Technology, Gaithersburg, MD, August 2015.

[5] _, *Federal Information Processing Standards Publication 186-4 Digital Signature Standard (DSS)*, U.S. National Institute of Standards and Technology, Gaithersburg, MD, July 2013.

[6] _, *Federal Information Processing Standards Publication 186-5 Digital Signature Standard (DSS).* U.S. National Institute of Standards and Technology, Gaithersburg, MD, 3 February 2023.

[7] _, *Federal Information Processing Standards Publication 197 Update 1: Advanced Encryption Standard (AES)*, U.S. National Institute of Standards and Technology, Gaithersburg, MD, 9 May 2023.

[8] _, *Federal Information Processing Standards Publication 198-1: The Keyed Hash Message Authentication Code (HMAC)*, U.S. National Institute of Standards and Technology, Gaithersburg, MD, July 2008.

[9] _, *Federal Information Processing Standards Publication 202 SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, U.S. National Institute of Standards and Technology, Gaithersburg, MD, August 2015.

[10] - _, *IEEE 1619-2018, IEEE Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices*, Institute of Electrical and Electronics Engineers (IEEE) Standards Association, 25 January 2019.

[11] _, *IEEE 1363a-2004, IEEE Standard Specifications for Public-Key Cryptography – Amendment 1: Additional Techniques*, Institute of Electrical and Electronics Engineers (IEEE) Standards Association, 2 September 2004.

[12] _, *ISO/IEC 9797-1:2011, Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher, Edition 2*, International Organization for Standardization and International Electrotechnical Commission, March 2011

[13] _, *ISO/IEC 9797-2:2021, Information Technology –Message Authentication Codes (MACs) – Part 2: Mechanisms Using a Dedicated Hash-Function, Edition 3*, International Organization for Standardization and International Electrotechnical Commission, June 2021

[14] _, *ISO/IEC 10116:2017, Information Technology – Security Techniques – Modes of Operation for an n-bit Block Cipher, Edition 4*, International Organization for Standardization and International Electrotechnical Commission, July 2017.

[15] _, *ISO/IEC 10118-3:2018, IT Security Techniques – Hash-Functions – Part 3: Dedicated Hash-Functions, Edition 4*, International Organization for Standardization and International Electrotechnical Commission, October 2018.

[16] _, *ISO/IEC 11770-6:2016, Information Technology – Security Techniques – Key Management – Part 6: Key Derivation*, International Organization for Standardization and International Electrotechnical Commission, October 2016.

[17] _, *ISO/IEC 14888-3:2018, Information Technology – Security Techniques – Digital signatures with Appendix – Part 3: Discrete Logarithm Based Mechanisms*, International Organization for Standardization and International Electrotechnical Commission, November 2018.

[18] _, *ISO/IEC 18031:2011, Information Technology – Security Techniques - Random Bit Generation, Edition 2*, International Organization for Standardization and International Electrotechnical Commission, November 2011.

[19] _, *ISO/IEC 18031:2011/Amd 1:2017, Information Technology – Security Techniques – Random Bit Generation – Amendment 1: Deterministic Random Bit Generation, Edition 2*, International Organization for Standardization and International Electrotechnical Commission, February 2017.

[20] _, *ISO/IEC 18033-2 :2006, Information Technology – Security Techniques – Encryption Algorithms – Part 2: Asymmetric Ciphers*, International Organization for Standardization and International Electrotechnical Commission, May 2006.

[21] _, *ISO/IEC 18033-3:2010, Information Technology – Security Techniques – Encryption Algorithms – Part 3: Block Ciphers Edition 2*, International Organization for Standardization and International Electrotechnical Commission, December 2010.

[22] _, *ISO/IEC 19772:2020, Information Technology – Authenticated Encryption, Edition 2*, International Organization for Standardization and International Electrotechnical Commission, November 2020.

[23] _, *ISO/IEC 29192-2:2019, Information Technology – Lightweight Cryptography – Part 2: Block Ciphers, Edition 2*, International Organization for Standardization and International Electrotechnical Commission, November 2019.

[24] E. Barker, *NIST Special Publication (SP) 800-57, Revision 5: Recommendation for Key Management: Part 1 – General*, National Institute of Standards and Technology, Gaithersburg, MD, May 2020.

[25] E. Barker, L. Chen, and R. Davis, *NIST Special Publication (SP) 800-56C Revision 2: Recommendation for Key-Derivation Methods in Key-Establishment Schemes*, National Institute of Standards and Technology, Gaithersburg, MD, August 2020.

[26] E. Barker, L. Chen, A. Roginsky, A. Vassilev, and R. Davis, *NIST Special Publication (SP) 800-56A Revision 3: Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*, National Institute of Standards and Technology, Gaithersburg, MD, April 2018.

[27] E. Barker, L. Chen, A. Roginsky, A. Vassilev, R. Davis, S. Simon, *NIST Special Publication (SP) 800-56B Revision 2: Recommendation for Pair-Wise Key Establishment Using Integer Factorization Cryptography*, National Institute of Standards and Technology, Gaithersburg, MD, March 2019.

[28] E. Barker and J. Kelsey, *NIST Special Publication (SP) 800-90A Revision 1: Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, National Institute of Standards and Technology, Gaithersburg, MD, June 2015.

[29] E. Barker and A. Roginsky, *NIST Special Publication (SP) 800-131A Revision 2: Transitioning the Use of Cryptographic Algorithms and Key Lengths*. National Institute of Standards and Technology, Gaithersburg, MD, March 2019.

[30] E. Barker, A. Roginsky, R. Davis, *NIST Special Publication (SP) 800-133 Revision 2: Recommendation for Cryptographic Key Generation*, National Institute of Standards and Technology, Gaithersburg, MD, June 2020.

[31] D. R. L. Brown, *Standards for Efficient Cryptography - SEC 1: Elliptic Curve Cryptography, Version 2.0,* Certicom Corp, 21 May 2009.

[32] L. Chen, *NIST Special Publication (SP) 800-108 Revision 1 Update 1: Recommendation for Key Derivation Using Pseudorandom Functions*, National Institute of Standards and Technology, Gaithersburg, MD, 2 February 2024.

[33] L. Chen, D. Moody, A. Regenscheid, A. Robinson, K. Randall, *NIST Special Publication (SP) 800-186: Recommendations for Discrete Logarithm-based Cryptography: Elliptic Curve Domain Parameters*. National Institute of Standards and Technology, Gaithersburg, MD, February 2023.

[34] D. A. Cooper, D. C. Apon, Q. H. Dang, M, S. Davidson, M. J. Dworkin, C. A. Miller, *NIST Special Publication (SP) 800-208: Recommendation for Stateful Hash-Based Signature Schemes*, National Institute of Standards and Technology, Gaithersburg, MD, October 2020.

[35] Q. Dang, *NIST Special Publication (SP) 800-135 Revision 1: Recommendation for Existing Application-Specific Key Derivation Functions*, National Institute of Standards and Technology, Gaithersburg, MD, December 2011.

[36] M. Dworkin, *NIST Special Publication (SP) 800-38A: Recommendation for Block Cipher Modes of Operation Methods and Techniques*, National Institute of Standards and Technology, Gaithersburg, MD, December 2001.

[37] M. Dworkin, *NIST Special Publication (SP) 800-38B Updated: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, National Institute of Standards and Technology, Gaithersburg, MD, 6 October 2016.

[38] M. Dworkin, *NIST Special Publication (SP) 800-38C Updated: Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*, National Institute of Standards and Technology, Gaithersburg, MD, 20 July 2007.

[39] M. Dworkin, *NIST Special Publication (SP) 800-38D: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, National Institute of Standards and Technology, Gaithersburg, MD, November 2007.

[40] M. Dworkin, *NIST Special Publication (SP) 800-38E: Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices*, National Institute of Standards and Technology, Gaithersburg, MD, January 2010.

[41] M. Dworkin, *NIST Special Publication (SP) 800-38F: Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping*, National Institute of Standards and Technology, Gaithersburg, MD, December 2012.

[42] D. Florencio and C. Herley, “A Large Scale Study of Web Password Habits.”, WWW 2007 , May 2007.

[43] D. K. Gillmor, *Negotiated Finite Field Diffie-Hellman Ephermeral Parameters for Transport Layer Security (TLS),* Internet Engineering Task Force, August 2016.

[44] P. A. Grassi, J. L. Fenton, E. M. Newton, R. A. Perlner, A. R. Regenscheid, W. E. Burr, J. P. Richer, N. B. Lefkovitz, J. M. Danker, Y. Choong, K. K. Greene, and M. F. Theofanos, *NIST Special Publication (SP) 800-63B: Digital Identity Guidelines: Authentication and Lifecycle Management*. National Institute of Standards and Technology, Gaithersburg, MD, June 2017.

[45] A. Huelsing, D. Butin, S. Gazdag, J. Rijneveld, and A. Mohaisen., *Request for Comments (RFC) 8391: XMSS: Extended Merkle Signature Scheme*, Internet Research Task Force (IRTF), May 2018.

[46] S. Josefsson. and I. Liusvaara, *Request for Comments (RFC) 8032. Edwards-Curve Digital Signature Algorithm (EdDSA)*, Internet Research Task Force (IRTF), January 2017.

[47] J. Kelsey, S. Chang, and R. Perlner *NIST Special Publication (SP) 800-185: SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash, and ParallelHash*, National Institute of Standards and Technology, Gaithersburg, MD, December 2016.

[48] *W. Killman and W. Schindler, Anwendungshinweise aund Interpretationen zum Schema (AIS), AIS-31 – A Proposal for: Functionality Classes for Random Number Generators, Version 2.0,* Bundesamt für Sicherheit in der Informationstechnik (BSI), 18 September 2011.

[49] M. Kojo and T. Kevinen, *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*, Internet Engineering Task Force, May 2003.

[50] A. Langley, M. Hamburg, and S. Turner, *Request for Comments (RFC) 7748: Elliptic Curves for Security*, Internet Research Task Force (IRTF), January 2016.

[51] M. Lochter, and J. Merkle, *Request for Comments (RFC) 5639*: *Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*, Independent Submission, March 2010.

[52] D. McGrew, M. Curcio, and S. Fluhrer, *Leighton-Micali Hash-Based Signatures*, Internet Engineering Task Force, April 2019.

[53] K. Moriary (Ed), B. Kaliski, J. Jonsson, and A. Rusch, *Request for Comments (RFC) 8017: PKCS #1: RSA Cryptography Specifications Version 2.2*, Internet Engineering Task Force (IETF). November 2016.

[54] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle, *NIST Special Publication (SP) 800-90B: Recommendation for the Entropy Sources Used for Random Bit Generation*, National Institute of Standards and Technology, Gaithersburg, MD, January 2018.