collaborative Protection Profile for Dedicated Security Component

Version: 1.0
2020-09-10
National Information Assurance Partnership

Revision History

Version	Date	Comment
1.0	2020-09-10	First published release version.
1.0x	2021-04-06	Start of first XML version.

Contents

1 PP introduction

1.1 PP Reference Identification

PP Reference: collaborative Protection Profile for Dedicated Security Component

PP Version: 1.0

PP Date: September 10, 2020

1.2 Overview

The scope of this Protection Profile (PP) is to describe the security functionality of QQQQ products in terms of [CC] and to define functional and assurance requirements for such products. An operating system is software that manages computer hardware and software resources, and provides common services for application programs. The hardware it manages may be physical, virtual or imaginary.
Something

This is going to show some tests:

- Terms with abbrs like ASLR, or API, should be found a linked automatically.
- And components can be refered to by their name: FQQ_QQQ.1
- And so can requirements: FQQ_QQQ.1.1 or by their unique identifier:
- Or you can stop them ASLR
- This is how you do a picture:

Figure 1: Niap's Logo

- And this is how you reference it: Figure 1
- This is how you do an equation with an arbitrary counter:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad (1)$$

- And this is how you reference it: Eq. 1
- The following content should be included if:
  - the TOE implements ""

  Some text
- The following content should be included if:
  - the TOE implements "Widget Thing"

  Someting dependent on a feature
- And here's the audit event table for mandatory requirements. Table t-audit-mandatory for more information.
- Test for an xref to section Section 3.1 Threats

And this is another sentence (or fragment). I added this sentence and deleted the next one. This uses the plural acronym OSes.

And here's a generic coutner Abc 1: Some Words

And here's the reference to it Abc 1.

1.3 Terms

The following sections list Common Criteria and technology terms used in this document.

1.3.1 Common Criteria Terms

Assurance	Grounds for confidence that a TOE meets the SFRs [CC].
Base Protection Profile (Base-PP)	Protection Profile used as a basis to build a PP-Configuration.
Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation (International Standard ISO/IEC 15408).
Common Criteria Testing	Within the context of the Common Criteria Evaluation and Validation Scheme (CCEVS), an IT security evaluation facility, accredited by the National Voluntary Laboratory Accreditation Program (NVLAP) and approved by the NIAP Validation Body to conduct

Laboratory	Common Criteria-based evaluations.
Common Evaluation Methodology (CEM)	Common Evaluation Methodology for Information Technology Security Evaluation.
Distributed TOE	A TOE composed of multiple components operating as a logical whole.
Operational Environment (OE)	Hardware and software that are outside the TOE boundary that support the TOE functionality and security policy.
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Protection Profile Configuration (PP-Configuration)	A comprehensive set of security requirements for a product type that consists of at least one Base-PP and at least one PP-Module.
Protection Profile Module (PP-Module)	An implementation-independent statement of security needs for a TOE type complementary to one or more Base Protection Profiles.
Security Assurance Requirement (SAR)	A requirement to assure the security of the TOE.
Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.
Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in an ST.
Target of Evaluation (TOE)	The product under evaluation.

1.3.2 Technical Terms

Address Space Layout Randomization (ASLR)	An anti-exploitation feature which loads memory mappings into unpredictable locations. ASLR makes it more difficult for an attacker to redirect control to code that they have introduced into the address space of a process.
Administrator	An administrator is responsible for management activities, including setting policies that are applied by the enterprise on the operating system. This administrator could be acting remotely through a management server, from which the system receives configuration policies. An administrator can enforce settings on the system which cannot be overridden by non-administrator users.
Application (app)	Software that runs on a platform and performs tasks on behalf of the user or owner of the platform, as well as its supporting documentation.
Application Programming Interface (API)	A specification of routines, data structures, object classes, and variables that allows an application to make use of services provided by another software component, such as a library. APIs are often provided for a set of libraries included with the platform.
Credential	Data that establishes the identity of a user, e.g. a cryptographic key or password.
Critical Security Parameters (CSP)	Information that is either user or system defined and is used to operate a cryptographic module in processing encryption functions including cryptographic keys and authentication data, such as passwords, the disclosure or modification of which can compromise the security of a cryptographic module or the security of the information protected by the module.
DAR Protection	Countermeasures that prevent attackers, even those with physical access, from extracting data from non-volatile storage. Common techniques include data encryption and wiping.
Data Execution Prevention (DEP)	An anti-exploitation feature of modern operating systems executing on modern computer hardware, which enforces a non-execute permission on pages of memory. DEP prevents pages of memory from containing both data and instructions, which makes it more difficult for an attacker to introduce and execute code.

Developer	An entity that writes OS software. For the purposes of this document, vendors and developers are the same.
General Purpose Operating System	A class of OSes designed to support a wide-variety of workloads consisting of many concurrent applications or services. Typical characteristics for OSes in this class include support for third-party applications, support for multiple users, and security separation between users and their respective resources. General Purpose Operating Systems also lack the real-time constraint that defines Real Time Operating Systems (RTOS). RTOSes typically power routers, switches, and embedded devices.
Host-based Firewall	A software-based firewall implementation running on the OS for filtering inbound and outbound network traffic to and from processes running on the OS.
Operating System (OS)	Software that manages physical and logical resources and provides services for applications. The terms *TOE* and *OS* are interchangeable in this document.
Personally Identifiable Information (PII)	Any information about an individual maintained by an agency, including, but not limited to, education, financial transactions, medical history, and criminal or employment history and information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records, etc., including any other personal information which is linked or linkable to an individual.[OMB]
Sensitive Data	Sensitive data may include all user or enterprise data or may be specific application data such as PII, emails, messaging, documents, calendar items, and contacts. Sensitive data must minimally include credentials and keys. Sensitive data shall be identified in the OS's TSS by the ST author.
User	A user is subject to configuration policies applied to the operating system by administrators. On some systems under certain configurations, a normal user can temporarily elevate privileges to that of an administrator. At that time, such a user should be considered an administrator.
Virtual Machine (VM)	Blah Blah Blah

1.4 Compliant Targets of Evaluation

1.4.1 TOE Boundary

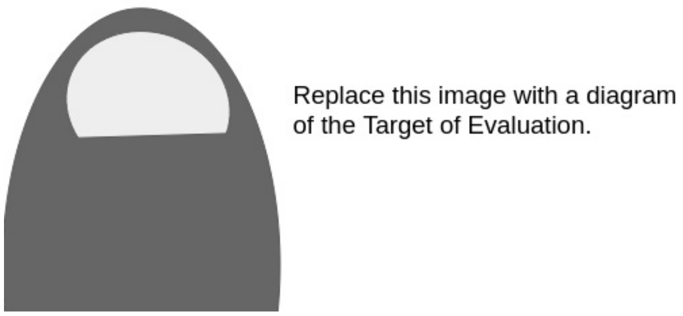

Figure 2: General TOE

1.4.2 TOE Platform

1.5 Use Cases

Requirements in this Protection Profile are designed to address the security problems in at least the following use cases. These use cases are intentionally very broad, as many specific use cases exist for an operating system. These use cases may also overlap with one another. An operating system's functionality may even be effectively extended by privileged applications installed onto it. However, these are out of scope of this PP.

[USE CASE 1] Elephant-own device

This is everything we need to describe in words about this use case.

For a the list of appropriate selections and acceptable assignment values for this configuration, see E.1 Elephant-own device.

2 Conformance Claims

Conformance Statement

An ST must claim exact conformance to this PP, as defined in the CC and CEM addenda for Exact Conformance, Selection-Based SFRs, and Optional SFRs (dated May 2017).

CC Conformance Claims

This PP is conformant to Parts 2 (extended) and 3 (conformant) of Common Criteria Version 3.1, Revision 5.

PP Claim

This PP does not claim conformance to any Protection Profile.

Package Claim

This PP is Functional Package for Transport Layer Security (TLS), version 1.1 Conformant and Functional Package for Secure Shell (SSH), version 1.0 Conformant .

3 Security Problem Description

The security problem is described in terms of the threats that the OS is expected to address, assumptions about the operational environment, and any organizational security policies that the OS is expected to enforce.

3.1 Threats

T.NETWORK_ATTACK
An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with applications and services running on or part of the OS with the intent of compromise. Engagement may consist of altering existing legitimate communications.

T.NETWORK_EAVESDROP
An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between applications and services that are running on or part of the OS.

T.LOCAL_ATTACK
An attacker may compromise applications running on the OS. The compromised application may provide maliciously formatted input to the OS through a variety of channels including unprivileged system calls and messaging via the file system.

T.LIMITED_PHYSICAL_ACCESS
An attacker may attempt to access data on the OS while having a limited amount of time with the physical device.

3.2 Assumptions

A.PLATFORM
The OS relies upon a trustworthy computing platform for its execution. This underlying platform is out of scope of this PP.

A.PROPER_USER
The user of the OS is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy. At the same time, malicious software could act *as* the user, so requirements which confine malicious subjects are still in scope.

A.PROPER_ADMIN
The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

4 Security Objectives

4.1 Security Objectives for the TOE

O.ACCOUNTABILITY
Conformant OSes ensure that information exists that allows administrators to discover unintentional issues with the configuration and operation of the operating system and discover its cause. Gathering event information and immediately transmitting it to another system can also enable incident response in the event of system compromise.

O.INTEGRITY
Conformant OSes ensure the integrity of their update packages. OSes are seldom if ever shipped without errors, and the ability to deploy patches and updates with integrity is critical to enterprise network security. Conformant OSes provide execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems.

O.MANAGEMENT
To facilitate management by users and the enterprise, conformant OSes provide consistent and supported interfaces for their security-relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration and application execution control.

O.PROTECTED_STORAGE
To address the issue of loss of confidentiality of credentials in the event of loss of physical control of the storage medium, conformant OSes provide data-at-rest protection for credentials. Conformant OSes also provide access controls which allow users to keep their files private from other users of the same system.

O.PROTECTED_COMMS
To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant OSes provide mechanisms to create trusted channels for CSP and sensitive data. Both CSP and sensitive data should not be exposed outside of the platform.

4.2 Security Objectives for the Operational Environment

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

OE.PLATFORM
The OS relies on being installed on trusted hardware.

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

OE.PROPER_ADMIN
The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

4.3 Security Objectives Rationale

This section describes how the assumptions, threats, and organization security policies map to the security objectives.

Table 1: Security Objectives Rationale

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

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

5 Security Requirements

This chapter describes the security requirements which have to be fulfilled by the product under evaluation. Those requirements comprise functional components from Part 2 and assurance components from Part 3 of [CC]. The following conventions are used for the completion of operations:

- **Refinement** operation (denoted by **bold text** or ~~strikethrough text~~): is used to add details to a requirement (including replacing an assignment with a more restrictive selection) or to remove part of the requirement that is made irrelevant through the completion of another operation, and thus further restricts a requirement.
- **Selection** (denoted by *italicized text*): is used to select one or more options provided by the [CC] in stating a requirement.
- **Assignment** operation (denoted by *italicized text*): is used to assign a specific value to an unspecified parameter, such as the length of a password. Showing the value in square brackets indicates assignment.
- **Iteration** operation: is indicated by appending the SFR name with a slash and unique identifier suggesting the purpose of the operation, e.g. "/EXAMPLE1."

5.1 Security Functional Requirements

5.1.1 Cryptographic Support (FCS)

FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1

The TSF shall generate cryptographic keys by [*parsing in accordance with FDP_ITC_EXT.1 and FDP_ITC_EXT.2*, [**selection**: *asymmetric key generation in accordance with FCS_CKM.1/AK, symmetric key generation in accordance to FCS_CKM.1/SK, no other methods*] ~~in accordance with a specified cryptographic key generation algorithm [assignment: *cryptographic key generation algorithm*] and specified cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*]~~.

Application Note: Parsing of keys can refer to both the act of importing keys from outside the TOE boundary and to the act of issuing commands or parameters to the TOE that trigger the TSF to perform a key generation function.

If asymmetric key generation in accordance with FCS_CKM.1/AK is selected, the selection-based SFR FCS_CKM.1/AK must be claimed by the TOE.

If symmetric key generation in accordance with FCS_CKM.1/SK is selected, the selection-based SFR FCS_CKM.1/SK must be claimed by the TOE.

Evaluation Activities

FCS_CKM.1:

FCS_CKM.1/AK Cryptographic Key Generation (Asymmetric Keys)

FCS_CKM.1.1/AK

The TSF shall generate **asymmetric** cryptographic keys using the methods defined by the following rows in Table 2: [**selection**: *AK1*, *AK2*, *AK3*, *AK4*, *AK5*].

Table 2: Supported Methods for Asymmetric Key Generation

Identifier	Key Type	Key Sizes	List of Standards
AK1	RSA	[**selection**: *2048 bit, 3072-bit*]	FIPS PUB 186-4 (Section B.3)
AK2	ECC-N	[**selection**: *256 (P-256), 384 (P-384), 521 (P-521)*]	FIPS PUB 186-4 (Section B.4 & D.1.2)
AK3	ECC-B	[**selection**: *256 (brainpoolP256r1), 384 (brainpoolP384r1), 512 (brainpoolP512r1)*]	RFC5639 (Section 3) (Brainpool Curves)
AK4	DSA	DSA Bit lengths of p and q respectively (L, N) [**selection**: *(1024, 160), (2048, 224), (2048, 256), (3027, 256)*]	FIPS 186-4 Appendix B.1
AK5	Curve25519	256 bits	RFC 7748

Application Note: This requirement is included for the purposes of encryption and decryption operations only. To support ITE protected communications requirement for the transfer of encrypted data, this requirement mandates implementation compliance to FIPS 186-4 only. Implementations according to

FIPS 186-2 or FIPS 186-3 will not be accepted.

This requirement must be claimed by the TOE if at least one of FCS_CKM.1 or FCS_CKM.1/KEK chooses a selection related to generation of asymmetric keys.

Evaluation Activities

FCS_CKM.1/AK:

FCS_CKM.1/SK Cryptographic Key Generation (Symmetric Encryption Key)

FCS_CKM.1.1/SK

The TSF shall generate **symmetric** cryptographic keys using the methods defined by the following rows in Table 3: [**selection**: *RSK, DSK, PBK*].

Table 3: Supported Methods for Symmetric Key Generation

Identifier	Key Type	Cryptographic Key Generation Algorithm	Key Sizes	List of Standards
RSK	[**selection**: *symmetric key, submask, authorization value*]	Direct Generation from a Random Bit Generator as specified in FCS_RBG_EXT.1	[**selection**: *128, 192, 256, 512*] bits	NIST SP 800-133 (Section 7.1) with ISO 18031 as an approved RBG in addition to those in NIST SP 800-133 (Section 5).
DSK [**selection**: *identifier from Table 16: Key Derivation Functions*]	[**selection**: *Key Type from Table 16: Key Derivation Functions*]	Derived from a Key Derivation Function as specified in FCS_CKM_EXT.5 [**selection**: *Key Derivation Algorithm from Table 16: Key Derivation Function*]	[**selection**: *key sizes from Table 16: Key Derivation Functions*]	[**selection**: *List of Standards from Table 16: Key Derivation Functions*]
PBK	[**selection**: *submask, authentication token, authorization value*]	Derived from a Password Based Key Derivation Function as specified in FCS_COP.1/PBKDF	[**selection**: *key sizes as specified in FCS_COP.1/PBKDF*]	[**selection**: *standards as specified in FCS_COP.1/PBKDF*]

Application Note: The intent of this requirement is to ensure that attackers cannot recover SKs with less than a full exhaust of the key space. This requirement explains SK generation regardless of how the DSC uses it or when it generates it. The encryption of user data that is not keying material, authentication tokens, or authorization values is outside the scope of this cPP. This cPP assumes that the DSC lacks the required resources to perform bulk encryption/decryption services at a suitable rate for users. The host may use the SK for encrypting user data outside the boundaries of the DSC. On the other hand, the DSC may use the SK on behalf of the user to perform keyed hashes. In this case, all the requirements for generating, controlling access and use, and destroying the key while under the protection of the DSC apply. The selection of key size 512 bits is for the case of XTS-AES using AES-256. In the case of XTSAES for both AES-128 and AES-256, the developer is expected to ensure that the full key is generated using direct generation from the RBG as in NIST SP 800-133 section.

The ST author selects at least one algorithm from the RSK row if the ST supports creating keys directly from the output of the RBG without further conditioning, at least one algorithm from the DSK row should be selected if the ST supports key derivation functions which are usually seeded from RBG and then further conditioned to the appropriate key size, and at least one algorithm from the PBK row should be selected if the ST supports keys derived from passwords.

If DSK is selected, the selection-based SFR FCS_CKM_EXT.5 must be claimed by the TOE.

If PBK is selected, the selection-based SFR FCS_COP.1/PBKDF must be claimed by the TOE.

This requirement must be claimed by the TOE if at least one of FCS_CKM.1 or FCS_CKM.1/KEK chooses a selection related to generation of symmetric keys.

Evaluation Activities

FCS_CKM.1/SK:

FCS_CKM.1/KEK Cryptographic Key Generation (Key Encryption Key)

FCS_CKM.1.1/KEK

The TSF shall generate key encryption keys in accordance with a specified cryptographic key generation algorithm corresponding to [**selection**:

- *Asymmetric KEKs generated in accordance with FCS_CKM.1/AK identifier AK1,*
- *Symmetric KEKs generated in accordance with FCS_CKM.1/SK,*
- *Derived KEKs generated in accordance with FCS_CKM_EXT.5*

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

Application Note: KEKs protect KEKs and Symmetric Keys (SKs). DSCs should use key strengths commensurate with protecting the chosen symmetric encryption key strengths. If Asymmetric KEKs generated in accordance with FCS_CKM.1/AK is selected, the selection-based SFR FCS_CKM.1/AK must be claimed by the TOE.

If Symmetric KEKs generated in accordance with FCS_CKM.1/SK is selected, the selection-based SFR FCS_CKM.1/SK must be claimed by the TOE.

If Derived KEKs generated in accordance with FCS_CKM_EXT.5 is selected, the selection-based SFR FCS_CKM_EXT.5 must be claimed by the TOE.

Evaluation Activities

FCS_CKM.1/KEK:
Test test test

FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1

The TSF shall establish cryptographic keys in accordance with a specified cryptographic key establishment method: [**selection**:

- *RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography”,*
- *RSA-based key establishment schemes that meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, “Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.2”,*
- *Elliptic curve-based key establishment schemes that meet the following: [**selection**:*
    - *NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”,*
    - *RFC 7748, “Elliptic Curves for Security”*

  ],
- *Finite field-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”,*
- *Elliptic Curve Integrated Encryption Scheme (ECIES) that meets the following: [**selection**:*
    - *ANSI X9.63 - Public Key Cryptography for the Financial Services Industry Key Agreement and Key Transport Using Elliptic Curve Cryptography,*
    - *IEEE 1363a - Standard Specification for Public-Key Cryptography - Amendment 1: Additional Techniques,*
    - *ISO/IEC 18033-2 - Information Technology - Security Techniques - Encryption Algorithms - Part 2: Asymmetric Ciphers,*
    - *SECG SEC1 - Standards for Efficient Cryptography Group Elliptic Curve Cryptography, section 5.1 Elliptic Curve Integrated Encryption Scheme*

  ]

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

Application Note: This is a refinement of the SFR FCS_CKM.2 to deal with key establishment rather than key distribution.

The ST author selects all key establishment schemes used for the selected cryptographic protocols.

The RSA-based key establishment schemes are described in Section 8 of NIST SP 800-56B Revision 2 [NIST-RSA]; however, Section 8 relies on implementation of other sections in SP 800-56B Revision 2.

The elliptic curves used for the key establishment scheme correlate with the curves specified in FCS_CKM.1/AK.

The selections in this SFR must be consistent with those for FCS_COP.1/KAT.

Evaluation Activities

FCS_CKM.2:

FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1

The TSF shall destroy cryptographic keys and keying material in accordance with a specified cryptographic key destruction method

- For volatile memory, the destruction shall be executed by a [**selection**:
  - *single overwrite consisting of [**selection**: a pseudo-random pattern using the TSF's RBG, zeroes, ones, a new value of a key, [**assignment**: some value that does not contain any CSP]],*
  - *removal of power to the memory,*
  - *removal of all references to the key directly followed by a request for garbage collection*

  ]
- For non-volatile memory [**selection**:
  - *that employs a wear-leveling algorithm, the destruction shall be executed by a [**selection**:*
    - *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]],*
    - *block erase*

    ],
  - *that does not employ a wear-leveling algorithm, the destruction shall be executed by a [**selection**:*
    - *[**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 again up to [**assignment**: number of times to attempt overwrite] times, whereupon an error is returned.,*
    - *block erase*

    ]

  ]

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

Application Note: A DSC must implement mechanisms to destroy cryptographic keys and key material contained in persistent storage when no longer needed. The term “cryptographic keys” in this SFR includes the authorization data that is the entry point to a key chain and all other cryptographic keys and keying material (whether in plaintext or encrypted form). 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.

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). Guidance documentation for the TOE requires users not to allow the TOE to leave the user’s control while a session is active (and hence while the DEK is likely to be in plaintext in volatile memory).

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 collaborative Protection Profile for Dedicated Security Components the mappings of logical addresses to the erased memory locations are erased as well as the data itself.

Where different destruction methods are used for different data or different destruction situations then the different methods and the data/situations they apply to (e.g. different points in time, or power-loss situations) are described in the TSS (and the ST may use separate iterations of the SFR to aid clarity). The TSS includes a table describing all relevant keys and keying material (including authorization data) used in the implementation of the SFRs, stating the source of the data, all memory types in which the data is stored (covering storage both during and outside of a session, and both plaintext and non-plaintext forms of the data), and the applicable destruction method and time of destruction in each case.

Some selections allow 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_EXT requirements, and not being any of the particular values listed as other selection options. The point of the phrase “does not contain any sensitive data” is to ensure that the overwritten data is carefully selected, and not taken from a general pool that might contain current or residual data (e.g. SDOs or intermediate key chain values) that itself requires confidentiality protection.

Evaluation Activities

FCS_CKM.4:

FCS_CKM_EXT.4 Cryptographic Key and Key Material Destruction Timing

FCS_CKM_EXT.4.1

The TSF shall destroy all keys and keying material when no longer needed.

Application Note: The DSC will have mechanisms to destroy keys, including intermediate keys and key material, by using an approved method, FCS_CKM.4. Examples of keys include intermediate keys, leaf keys, encryption keys, signing keys, verification keys, authentication tokens, and submasks. The DSC will have mechanisms to destroy keys and key material contained in persistent storage when no longer needed. Based on their implementation, vendors will explain when certain keys are no longer needed. An example in which key is no longer necessary includes a wrapped key whose password has changed. However, there are instances when keys are allowed to remain in memory, for example, a device identification key.

Evaluation Activities

FCS_CKM_EXT.4:

FCS_CKM_EXT.5 Cryptographic Key Derivation

FCS_CKM_EXT.5.1

The TSF shall generate cryptographic keys using the Key Derivation Functions defined by the following rows of Table 4: [**selection**: *KeyDrv1*, *KeyDrv2*, *KeyDrv3*, *KeyDrv4*, *KeyDrv5*, *KeyDrv6*, *KeyDrv7*, *KeyDrv8*].

Table 4: Key Derivation Functions

Identifier	Key Type	Input Parameters	Key Derivation Algorithm	Key Sizes	List of Stan
KeyDrv1	[**selection**: *symmetric key, initialization vector, authentication token, authorization value, HMAC key, KMAC key*]	Direct Generation from a Random Bit Generator as specified in FCS_RBG_EXT.1	KDF in Counter Mode using [**selection**: *CMAC-AES-128, CMAC-AES-192, CMAC-AES-256, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512*]as the PRF	[**selection**: *128, 192, 256*]bits	NIST SP 800 (Section 5.1 in Counter M [**selection**: *CMAC, NIST CMAC, ISO- ISO-HMAC, HMAC, ISO- FIPS-SHA*]
KeyDrv2	[**selection**: *symmetric key, initialization vector, authentication token, authorization value, HMAC key, KMAC key*]	Direct Generation from a Random Bit Generator as specified in FCS_RBG_EXT.1	KDF in Feedback Mode using [**selection**: *CMAC-AES-128, CMAC-AES-192, CMAC-AES-256, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512*]as the PRF	[**selection**: *128, 192, 256*]bits	NIST SP 800 (Section 5.2 in Feedback [**selection**: *CMAC, NIST CMAC, ISO- ISO-HMAC, HMAC, ISO- FIPS-SHA*]
KeyDrv3	[**selection**: *symmetric key, initialization vector, authentication token, authorization value, HMAC key, KMAC key*]	Direct Generation from a Random Bit Generator as specified in FCS_RBG_EXT.1	KDF in Double Pipeline Iteration Mode using [**selection**: *CMAC-AES-128, CMAC-AES-192, CMAC-AES-256, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512*]as the PRF	[**selection**: *128, 192, 256*]bits	NIST SP 800 (Section 5.3 in n Double Pipeline Iter Mode) [**selection**: *CMAC, NIST CMAC, ISO- ISO-HMAC, HMAC, ISO- FIPS-SHA*]
KeyDrv4	[**selection**: *symmetric key, initialization vector, authentication token, authorization*	Intermediary keys	[**selection**: *exclusive OR (XOR), SHA256, SHA-512*]	[**selection**: *128, 192, 256*]bits	[**selection**: *HASH, FIPS*

	value, HMAC key, KMAC key]				
KeyDrv5	[**selection**: *symmetric key, initialization vector, authentication token, authorization value, HMAC key, KMAC key*]	Concatenated keys	KDF in [**selection**: *Counter Mode, Feedback Mode, Double Pipeline Iteration Mode*] using [**selection**: *CMAC-AES-128, CMAC-AES-192, CMAC-AES-256, HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512*]as the PRF	[**selection**: *128, 192, 256*]bits	NIST SP 800 [selection: (S 5.1) (KDF in Counter Mo (Section 5.2) in Feedback Mode); (Section 5.3) in Double-Pi Iteration Mo [**selection**: *CMAC, NIST CMAC, ISO- ISO-HMAC, HMAC, ISO- FIPS-SHA*]
KeyDrv6	[**selection**: *symmetric key, initialization vector, authentication token, authorization value, HMAC key, KMAC key*]	Two keys	[**selection**: *AES-CCM, AES-GCM, AES-CBC, AES-KWP, AES-KW, CAM-CBC, CAM-CCM, CAM-GCM*] from FCS_COP.1/SKC Symmetric Key table	[**selection**: *128, 192, 256*]bits	[selection: se of Standards FCS_COP.1/ Symmetric K table]
KeyDrv7	[**selection**: *symmetric key, secret IV, seed*]	Shared secret, salt, output length, fixed information	[**selection**: *hash function from FCS_COP.1/Hash, keyed hash from FCS_COP.1/HMAC*]	[**selection**: *128, 192, 256*]bits	(NIST-KDRV [**selection**: *List of Stand in FCS_COP.1/ and FCS_COP.1/*
KeyDrv8	[**selection**: *symmetric key, secret IV, seed*]	Shared secret, salt, IV, output length, fixed information	[**selection**: *keyed hash from FCS_COP.1/HMAC*]	[**selection**: *128, 192, 256*]bits	(NIST-KDRV [**selection**: *List of Stand in FCS_COP.1/ and FCS_COP.1/*

Application Note: Note that Camellia algorithms do not support 192-bit key sizes. The interface referenced in the requirement could take different forms, the most likely of which is an application programming interface to an OS kernel. There may be various levels of abstraction. For Authorization Factor Submasks, the key size to be used in the HMAC falls into a range between L1 and L2 defined in ISO/IEC 10118 for the appropriate hash function (for example for SHA-256 L1 = 512, L2 =256) where L2 = k = L1.

General note: in order to use a NIST SP 800-108 conformant method of key derivation, the TOE is permitted to implement this with keys as derived as indicated in Key Derivation Functions table above, and with the algorithms as indicated in the same table.

NIST SP 800-131A Rev 1 allows the use of SHA-1 in these use cases.

KeyDrv5, KeyDrv6, and the XOR option in KeyDrv4 will create an “inverted key hierarchy” in which the TSF will combine two or more keys to create a third key. These same KDFs may also use a submask key as input, which could be an authorization factor or derived from a PBKDF. In these cases the ST author must explicitly declare this option and should present a reasonable argument that the entropy of the inputs to the KDFs will result in full entropy of the expected output.

If keys are combined, the ST author shall describe which method of combination is used in order to justify that the effective entropy of each factor is preserved.

The documentation of the product’s encryption key management should be detailed enough that, after reading, the evaluator will thoroughly understand the product’s key management and how it meets the requirements to ensure the keys are adequately protected. This documentation should include an essay and diagrams. This documentation is not required to be part of the TSS; it can be submitted as a separate document and marked as developer proprietary.

SP 800-56C specifies a two-step key derivation procedure that employs an extraction-thenexpansion technique for deriving keying material from a shared secret generated during a key establishment scheme. The Randomness

Extraction step as described in Section 5 of SP 800-56C is followed by Key Expansion using the key derivation functions defined in SP 800-108.

This requirement must be claimed by the TOE if at least one of FCS_CKM.1/KEK, FCS_CKM.1/SK, or FCS_COP.1/KeyEnc chooses a selection related to key derivation.

If at least one of KeyDrv4, KeyDrv5, or KeyDrv6 is selected AND password-based key derivation is used to create at least one of the inputs, the selection-based SFR FCS_COP.1/PBKDF must also be claimed.

Evaluation Activities

FCS_CKM_EXT.5:

FCS_COP.1/Hash Cryptographic Operation (Hashing)

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*, *SHA3-224*, *SHA3-256*, *SHA3-384*, *SHA3-512*] that meets the following: [**selection**: *ISO/IEC 10118-3:2018*, *FIPS 180-4*].

Application Note: The hash selection should be consistent with the overall strength of the algorithm used for signature generation. For example, the DSC 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 for the following applications: generating and verifying hash-based message authentication codes (HMACs), key derivation functions (KDFs), and random bit/number generation (In certain cases, SHA-1 may also be used for verifying old digital signatures and time stamps, provided that this is explicitly allowed by the application domain).

Evaluation Activities

FCS_COP.1/Hash:
Nothing to see here.

FCS_COP.1/HMAC Cryptographic Operation (Keyed Hash)

FCS_COP.1.1/HMAC

The TSF shall perform [*keyed hash message authentication*] in accordance with a specified cryptographic algorithm [**selection**: *HMAC-SHA-1*, *HMAC-SHA-256*, *HMAC-SHA-384*, *HMAC-SHA-512*, *KMAC128*, *KMAC256*] and cryptographic key sizes [**assignment**: *key size (in bits)*] that meet the following: [**selection**: *ISO/IEC 9797-2:2011 Section 7 "MAC Algorithm 2"*, *[NIST-KDV] section 4 "KMAC"*].

Application Note: The HMAC key size falls into a range between L1 and L2 defined in ISO/IEC 10118 for the appropriate hash function (for example for SHA-256 L1 = 512, L2 = 256) where L2 ≤ k ≤ L1.

Evaluation Activities

FCS_COP.1/HMAC:

FCS_COP.1/KAT Cryptographic Operation (Key Agreement/Transport)

FCS_COP.1.1/KAT

The TSF shall perform [*cryptographic key agreement/transport*] using the supported methods for key agreement/transport defined by the following rows of Table 5: [**selection**: *KAS1*, *KAS2*, *KTS-OAEP*, *RSAES-PKCS1-v1_5*, *ECDH-NIST*, *ECDH-BPC*, *DH*, *Curve25519*, *ECIES*].

Table 5: Supported Methods for Key Agreement/Transport Operation

Identifier	Cryptographic Algorithm	Key Sizes	List of Standards
KAS1	RSA-single party	[**selection**: *2048*, *3072*, *4096*, *6144*, *8192*]bits	NIST SP 800-56Br2 section 8.2
KAS2	RSA-both party	[**selection**: *2048*, *3072*, *4096*, *6144*, *8192*]bits	NIST SP 800-56Br2 section 8.3
KTS-OAEP	RSA	[**selection**: *2048*,	NIST SP 800-56Br2

		3072, *4096*, *6144*, *8192*]bits	section 9
RSAES-PKCS1-v1_5	RSA	[**selection**: *2048*, *3072*, *4096*, *6144*, *8192*]bits	RFC 8017 Section 7.2
ECDH-NIST	ECDH with NIST curves	[**selection**: *256 (P-256)*, *384 (P-384)*, *512 (P-521)*]	NIST SP 800-56Ar3
ECDH-BPC	ECDH with Brainpool curves	[**selection**: *256 (brainpoolP256r1)*, *384 (brainpoolP384r1*, *512 (brainpoolP512r1)*]	RFC 5639 (Section 3)
DH	Diffie-Hellman	[**selection**: *2048*, *3072*, *4096*, *6144*, *8192*]bits	NIST SP 800-56A rev 3, [**selection**: • *RFC 3526 Section [**selection**: 3, 4, 5, 6, 7],* • *RFC 7919 Appendices [**selection**: A.1, A.2, A.3, A.4, A.5]*]
Curve25519	ECDH	256 bits	RFC 7748
ECIES	ECIES	[**selection**: *256*, *384*, *512*]bits	[**selection**: *ANSI X9.63*, *IEEE 1363a*, *ISO/IEC 18033-2 Part 2*, *SECG SEC1 sec 5.1*]

Application Note: The selections in this SFR should be consistent with the algorithms selected in FCS_CKM.2.

Evaluation Activities

FCS_COP.1/KAT:

FCS_COP.1/KeyEnc Cryptographic Operation (Key Encryption)

FCS_COP.1.1/KeyEnc

The TSF shall perform [*key encryption and decryption*] using the methods defined in the following rows of Table 6: [**selection**: *SE1*, *AE1*, *SE2*, *XOR*]

Table 6: Supported Methods for Key Encryption Operation

Identifier	**Cryptographic Algorithm**	**Key Sizes**	**List of Standards**
SE1	Symmetric [**selection**: *AES-CCM*, *AES-GCM*, *AES-CBC*, *AES-CTR*, *AES-KWP*, *AESKW*]	[**selection**: *128*, *192*, *256*] bits	See FCS_COP.1/SKC
AE1	Asymmetric KTS-OAEP	[**selection**: *2048*, *3072*] bits	See FCS_COP.1/SKC
SE2	Symmetric [**selection**: *CAM-CBC*, *CAM-CCM*, *CAM-GCM*]	[**selection**: *128*, *256*] bits	See FCS_COP.1/KAT
XOR	Exclusive OR operation	[**selection**: *128*, *192*, *256*] bits	See FCS_CKM_EXT.5

Application Note: A TOE will use this requirement to specify how the Key Encryption Key (KEK) wraps a symmetric encryption key. A TOE will always need this requirement in order to capture the last stage of the key chain in which the Key Encryption Key (KEK) wraps the symmetric encryption key.

If XOR is selected, the selection-based SFR FCS_CKM_EXT.5 must be claimed by the TOE.

Evaluation Activities

FCS_COP.1/KeyEnc:

FCS_COP.1/SigGen Cryptographic Operation (Signature Generation)

FCS_COP.1.1/SigGen

The TSF shall perform [*digital signature generation*] using the supported methods for signature generation defined in the following rows of Table 7 [**selection**: *SigGen1*, *SigGen2*, *SigGen3*, *SigGen4*, *SigGen5*].

Table 7: Supported Methods for Signature Generation Operation

Identifier	Cryptographic Algorithm	Key Sizes	List of Standards
SigGen1	RSASSA-PKCS1-v1_5 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	[**selection**: *RFC 8017*, *PKCS #1 v2.2 (Section 8.2)*, *FIPS186-4*, *(Section 5.5)*] (RSASSA-PKCS1-v1_5) [**selection**: *ISO10118-3 (Clause 10, 11)*, *FIPS180-4 (Section 6)*] (SHA)
SigGen2	Digital signature scheme 2 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	ISO9796-2, (Clause 9) (Digital signature scheme 2) [**selection**: *ISO10118-3 (Clause 10, 11)*, *FIPS180-4 (Section 6)*] (SHA)
SigGen3	Digital signature scheme 3 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	ISO9796-2, (Clause 10) (Digital signature scheme 3) [**selection**: *ISO10118-3 (Clause 10, 11)*, *FIPS180-4 (Section 6)*] (SHA)
SigGen4	RSASSA-PSS using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	[RFC8017, PKCS#1v2.2 (Section 8.1)] (RSASSAPSS) [**selection**: *ISO10118-3 (Clause 10, 11)*, *FIPS180-4 (Section 6)*] (SHA)
SigGen5	ECDSA on [**selection**: *brainpoolP256r1*, *brainpoolP384r1*, *brainpoolP512r1*, *NIST P-256*, *NIST P-384*, *NIST P-521*] using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	[**selection**: • *[**selection**: ISO14888-3, FIPS186-4 (Section 6)] (EDCSA)*, • *RFC5639 (Section 3) (Brainpool Curves)*, • *FIPS186-4 (Appendix D.1.2) (NIST Curves)*

			] [**selection**: *ISO10118-3 (Clause 10, 11), FIPS180-4 (Section 6)*] (SHA)

Evaluation Activities

FCS_COP.1/SigGen:

FCS_COP.1/SigVer Cryptographic Operation (Signature Verification)

FCS_COP.1.1/SigVer

The TSF shall perform [*digital signature verification*] using the supported methods for signature verification defined in the following rows of Table 8 [**selection**: *SigVer1*, *SigVer2*, *SigVer3*, *SigVer4*, *SigVer5*].

Table 8: Supported Methods for Signature Verification Operation

Identifier	Cryptographic Algorithm	Key Sizes	List of Standards
SigVer1	RSASSA-PKCS1-v1_5 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	[**selection**: *RFC 8017, PKCS #1 v2.2 (Section 8.2), FIPS186-4, (Section 5.5)*] (RSASSA-PKCS1-v1_5) [**selection**: *ISO10118-3 (Clause 10, 11), FIPS180-4 (Section 6)*] (SHA)
SigVer2	Digital signature scheme 2 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	ISO9796-2, (Clause 9) (Digital signature scheme 2) [**selection**: *ISO10118-3 (Clause 10, 11), FIPS180-4 (Section 6)*] (SHA)
SigVer3	Digital signature scheme 3 using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	ISO9796-2, (Clause 10) (Digital signature scheme 3) [**selection**: *ISO10118-3 (Clause 10, 11), FIPS180-4 (Section 6)*] (SHA)
SigVer4	RSASSA-PSS using [**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]	[**selection**: *2048 bit*, *3072 bit*]	[RFC8017, PKCS#1v2.2 (Section 8.1)] (RSASSAPSS) [**selection**: *ISO10118-3 (Clause 10, 11), FIPS180-4 (Section 6)*] (SHA)
SigVer5	ECDSA on [**selection**: *brainpoolP256r1*, *brainpoolP384r1*, *brainpoolP512r1*, *NIST P-256*, *NIST P-384*, *NIST P-521*] using	[**selection**: *2048 bit*, *3072 bit*]	[**selection**: • *[**selection**: ISO14888-3, FIPS186-4*

	[**selection**: *SHA-256*, *SHA-384*, *SHA-512*, *SHA3-256*, *SHA3-384*, *SHA3-512*]		*(Section 6)]* *(EDCSA)*, • *RFC5639 (Section 3) (Brainpool Curves)*, • *FIPS186-4 (Appendix D.1.2) (NIST Curves)*] [**selection**: *ISO10118-3 (Clause 10, 11)*, *FIPS180-4 (Section 6)*] (SHA)

Evaluation Activities

FCS_COP.1/SigVer:

FCS_COP.1/SKC Cryptographic Operation (Symmetric Key Cryptography)

FCS_COP.1.1/SKC

The TSF shall perform [*data encryption/decryption*] using the supported symmetric-key cryptography methods defined in the following rows of Table 9 [**selection**: *AES-CCM*, *AES-GCM*, *AES-CBC*, *AES-CTR*, *XTS-AES*, *AES-KWP*, *AES-KW*, *CAM-CBC*, *CAM-CCM*, *CAM-GCM*, *XTS-CAM*].

Table 9: Supported Methods for Symmetric Key Cryptography Operation

Identifier	Cryptographic Algorithm	Key Sizes	List of Standards
AES-CCM	AES in CCM mode with unpredictable, nonrepeating nonce, minimum size of 64 bits	[**selection**: *128 bits*, *192 bits*, *256 bits*]	ISO 18033-3 (AES) ISO 19772, Clause 8 (CCM) NIST SP800-38C (CCM)
AES-GCM	AES 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 [**selection**: *96*, *104*, *112*, *120*, *128*]	[**selection**: *128 bits*, *192 bits*, *256 bits*]	ISO 18033-3 (AES) ISO 19772, Clause 11 (GCM) NIST SP800-38D (GCM)
AES-CBC	AES in CBC mode with non-repeating and unpredictable IVs	[**selection**: *128 bits*, *192 bits*, *256 bits*]	ISO 18033-3 (AES) ISO 10116 (CBC) NIST SP800-38A (CBC)
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 bits*, *192 bits*, *256 bits*]	ISO 18033-3 (AES) ISO 10116 (CTR) NIST SP800-38A (CTR)
XTS-AES	AES in XTS mode with unique [**selection**: *consecutive non-negative integers starting at an arbitrary non-negative integer*, *data unit sequence numbers*] tweak values	[**selection**: *256 bits*, *512 bits*]	ISO 18033-3 (AES) [**selection**: *IEEE 1619*, *NIST*

			SP800-38E](XTS)
AES-KWP	KWP based on AES	[**selection**: *128 bits, 192 bits, 256 bits*]	ISO 18033-3 (AES) NIST SP 800-38F, sec. 6.3 (KWP)
AES-KW	KW based on AES	[**selection**: *128 bits, 192 bits, 256 bits*]	ISO 18033-3 (AES) NIST SP 800-38F, sec. 6.2 (KW) ISO/IEC 19772, clause 7 (key wrap)
CAM-CBC	Camellia in CBC mode with non-repeating and unpredictable IVs	[**selection**: *128 bits, 256 bits*]	ISO 18033-3 (Camellia) ISO 10116 (CBC)
CAM-CCM	Camellia in CCM mode with unpredictable, nonrepeating nonce, minimum size of 64 bits	[**selection**: *128 bits, 256 bits*]	ISO 18033-3 (Camellia) ISO 19772, Clause 8 (CCM) NIST SP800-38C (CCM)
CAM-GCM	Camellia 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 [**selection**: *96, 104, 112, 120, 128*]	[**selection**: *128 bits, 256 bits*]	ISO 18033-3 (Camellia) ISO 19772, Clause 11 (GCM) NIST SP800-38D (GCM)
XTS-CAM	Camellia in XTS mode with unique [**selection**: *consecutive non-negative integers starting at an arbitrary non-negative integer, data unit sequence numbers*] tweak values	[**selection**: *256 bits, 512 bits*]	ISO 18033-3 (Camellia) [**selection**: *IEEE 1619, NIST SP800-38E*](XTS)

Evaluation Activities

FCS_COP.1/SKC:

FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1

The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [**selection**: *Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)*].

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded by at least one entropy source in accordance with NIST SP 800-90B that accumulates entropy from [**selection**: *[**assignment**: number of software-based sources] software-based noise source, [**assignment**: number of hardware-based sources] hardware-based noise source*] with a minimum of [**selection**: *128, 192, 256*] bits of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011, of the keys and CSPs that it will generate.

Application Note: ISO/IEC 18031:2011 contains three different methods of generating random numbers. Each of these in turn depends on underlying cryptographic primitives (hash functions/ciphers). This cPP allows SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 for Hash_DRBG or HMAC_DRBG and only AES-based implementations for CTR_DRBG.

Evaluation Activities

FCS_RBG_EXT.1:

FCS_SLT_EXT.1 Cryptographic Salt Generation

FCS_SLT_EXT.1.1

The TSF shall use salts and nonces generated by an RBG as specified in FCS_RBG_EXT.1.

Evaluation Activities

FCS_SLT_EXT.1:

FCS_STG_EXT.1 Protected Storage

FCS_STG_EXT.1.1

The TSF shall provide [**selection**: *mutable hardware-based*, *immutable hardware-based*, *software-based*] protected storage for asymmetric private keys and [**selection**: *symmetric keys*, *persistent secrets*, *no other keys*].

Application Note: If the protected storage is implemented in software that is protected as required by FCS_STG_EXT.2, the ST author is expected to select "software-based." If "software-based" is selected, the ST author is expected to select all "software-based key storage" in FCS_STG_EXT.2.

Support for protected storage for all symmetric keys and persistent secrets will be required in future revisions.

FCS_STG_EXT.1.2

FCS_STG_EXT.1.2 The TSF shall support the capability of [**selection**: *importing keys/secrets into the TOE*, *causing the TOE to generate keys/secrets*] upon request of [**selection**: *a client application*, *an administrator*].

FCS_STG_EXT.1.3

The TSF shall be capable of destroying keys/secrets in the protected storage upon request of [**selection**: *a client application*, *an administrator*].

FCS_STG_EXT.1.4

The TSF shall have the capability to allow only the user that [**selection**: *imported the key/secret*, *caused the key/secret to be generated*] to use the key/secret. Exceptions may be explicitly authorized only by [**selection**: *the client application*, *the administrator*].

FCS_STG_EXT.1.5

The TSF shall allow only the user that [**selection**: *imported the key/secret*, *caused the key/secret to be generated*] to request that the key/secret be destroyed. Exceptions may only be explicitly authorized by [**selection**: *the client application*, *the administrator*].

Application Note: Not all conformant TOEs will have the ability to import pre-generated keys into the TOE. In these cases, the TOE’s ability to receive commands to perform key generation is considered to be its implementation of the Parse service. A subject that caused a key to be generated is considered to be the ‘owner’ of that key in the same manner as they would be if they had imported it directly.

Evaluation Activities

FCS_STG_EXT.1:

FCS_STG_EXT.2 Key Storage Encryption

FCS_STG_EXT.2.1

The TSF shall encrypt [*AKs, SKs, KEKs, and [**selection**: long-term trusted channel key material, all software-based key storage, no other keys]*] using one of the following methods: [**assignment**: *key encryption methods as specified in FCS_COP.1/KeyEnc*].

Evaluation Activities

FCS_STG_EXT.2:

FCS_STG_EXT.3 Key Integrity Protection

FCS_STG_EXT.3.1

The TSF shall protect the integrity of any encrypted [*AKs, SKs, KEKs, and* [**selection**: *long-term trusted channel key material, all software-based key storage, no other keys*]] by using [**selection**:

- *Symmetric encryption in* [**selection**: *AES_CCM, AES_GCM, AES_KW, AES_KWP, CAM_CCM, CAM_GCM*] *mode in accordance with FCS_COP.1/SKC*,
- *A hash of the stored key in accordance with FCS_COP.1/Hash*,
- *A keyed hash of the stored key in accordance with FCS_COP.1/HMAC*,
- *A digital signature of the stored key in accordance with FCS_COP.1/SigGen using an asymmetric key that is protected in accordance with FCS_STG_EXT.2*,
- *An immediate application of the key for decrypting the protected data followed by a successful verification of the decrypted data with previously known information*

].

FCS_STG_EXT.3.2

The TSF shall verify the integrity of the [**selection**: *hash*, *digital signature*, *MAC*] of the stored key prior to use of the key.

Application Note: This requirement is not applicable to derived keys that are not stored. It is not expected that a single key will be protected from corruption by multiple of these methods; however, a product may use one integrity-protection method for one type of key and a different method for other types of keys. The documentation of the product’s encryption key management should be detailed enough that, after reading, the evaluator will thoroughly understand the product’s key management and how it meets the requirements to ensure the keys are adequately protected. This documentation should include an essay and diagrams. This documentation is not required to be part of the TSS – it can be submitted as a separate document and marked as developer proprietary.

Evaluation Activities

FCS_STG_EXT.3:

5.1.2 TOE Security Functional Requirements Rationale

The following rationale provides justification for each security objective for the TOE, showing that the SFRs are suitable to meet and achieve the security objectives:

Table 10: SFR Rationale

OBJECTIVE	ADDRESSED BY	RATIONALE
O.ACCOUNTABILITY	FAU_GEN.1	'cause FAU_GEN.1 is awesome
	FTP_ITC_EXT.1	Cause FTP reasons
O.INTEGRITY	FPT_SBOP_EXT.1	For reasons
	FPT_ASLR_EXT.1	ASLR For reasons
	FPT_TUD_EXT.1	For reasons
	FPT_TUD_EXT.2	For reasons
	FCS_COP.1/HASH	For reasons
	FCS_COP.1/SIGN	For reasons
	FCS_COP.1/KEYHMAC	For reasons
	FPT_ACF_EXT.1	For reasons
	FPT_SRP_EXT.1	For reasons
	FIA_X509_EXT.1	For reasons
	FPT_TST_EXT.1	For reasons
	FTP_ITC_EXT.1	For reasons
	FPT_W^X_EXT.1	For reasons
	FIA_AFL.1	For reasons
	FIA_UAU.5	For reasons
O.MANAGEMENT	FMT_MOF_EXT.1	For reasons
	FMT_SMF_EXT.1	For reasons

	FTA_TAB.1	For reasons
	FTP_TRP.1	For reasons
O.PROTECTED_STORAGE	FCS_STO_EXT.1, FCS_RBG_EXT.1, FCS_COP.1/ENCRYPT, FDP_ACF_EXT.1	Rationale for a big chunk
O.PROTECTED_COMMS	FCS_RBG_EXT.1, FCS_CKM.1, FCS_CKM.2, FCS_CKM_EXT.4, FCS_COP.1/ENCRYPT, FCS_COP.1/HASH, FCS_COP.1/SIGN, FCS_COP.1/HMAC, FDP_IFC_EXT.1, FIA_X509_EXT.1, FIA_X509_EXT.2, FTP_ITC_EXT.1	Rationale for a big chunk

5.2 Security Assurance Requirements

The Security Objectives in Section 4 Security Objectives were constructed to address threats identified in Section 3.1 Threats. The Security Functional Requirements (SFRs) in Section 5.1 Security Functional Requirements are a formal instantiation of the Security Objectives. The PP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.
This section lists the set of SARs from CC part 3 that are required in evaluations against this PP. Individual Assurance Activities o be performed are specified both in Section 5.1 Security Functional Requirements as well as in this section.
The general model for evaluation of OSs against STs written to conform to this PP is as follows:
After the ST has been approved for evaluation, the TSEF will obtain the OS, supporting environmental IT, and the administrative/user guides for the OS. The ITSEF is expected to perform actions mandated by the Common Evaluation Methodology (CEM) for the ASE and ALC SARs. The ITSEF also performs the Assurance Activities contained within Section 5.1 Security Functional Requirements, which are intended to be an interpretation of the other CEM assurance requirements as they apply to the specific technology instantiated in the OS. The Assurance Activities that are captured in Section 5.1 Security Functional Requirements also provide clarification as to what the developer needs to provide to demonstrate the OS is compliant with the PP.

5.2.1 Class ASE: Security Target

As per ASE activities defined in [CEM].

5.2.2 Class ADV: Development

The information about the OS is contained in the guidance documentation available to the end user as well as the TSS portion of the ST. The OS developer must concur with the description of the product that is contained in the TSS as it relates to the functional requirements. The Assurance Activities contained in Section 5.1 Security Functional Requirements should provide the ST authors with sufficient information to determine the appropriate content for the TSS section.

ADV_FSP.1 Basic Functional Specification (ADV_FSP.1)

The functional specification describes the TSFIs. It is not necessary to have a formal or complete specification of these interfaces. Additionally, because OSs conforming to this PP will necessarily have interfaces to the Operational Environment that are not directly invokable by OS users, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this PP, the activities for this family should focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional “functional specification” documentation is necessary to satisfy the assurance activities specified. The interfaces that need to be evaluated are characterized through the information needed to perform the assurance activities listed, rather than as an independent, abstract list.

Developer action elements:

ADV_FSP.1.1D

The developer shall provide a functional specification.

Content and presentation elements:

ADV_FSP.1.2C

The developer shall provide a tracing from the functional specification to the SFRs.

Application Note: As indicated in the introduction to this section, the functional specification is comprised of the information contained in the AGD_OPE and AGD_PRE documentation. The developer may reference a website accessible to application developers and the evaluator. The assurance activities in the functional requirements point to evidence that should exist in the documentation and TSS section; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

ADV_FSP.1.3C

The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.4C

The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.5C

The functional specification shall provide rationale for the implicit categorization of interfaces as SFR-non-interfering.

ADV_FSP.1.6C

The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements:

ADV_FSP.1.7E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.8E

The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

Evaluation Activities

ADV_FSP.1:
There are no specific assurance activities associated with these SARs, except ensuring the information is provided. The functional specification documentation is provided to support the evaluation activities described in Section 5.1 Security Functional Requirements, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other assurance activities being performed; if the evaluator is unable to perform an activity because there is insufficient interface information, then an adequate functional specification has not been provided.

5.2.3 Class AGD: Guidance Documentation

The guidance documents will be provided with the ST. Guidance must include a description of how the IT personnel verifies that the Operational Environment can fulfill its role for the security functionality. The documentation should be in an informal style and readable by the IT personnel. Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes instructions to successfully install the TSF in that environment; and Instructions to manage the security of the TSF as a product and as a component of the larger operational environment. Guidance pertaining to particular security functionality is also provided; requirements on such guidance are contained in the assurance activities specified with each requirement.

AGD_OPE.1 Operational User Guidance (AGD_OPE.1)

Developer action elements:

AGD_OPE.1.1D

The developer shall provide operational user guidance.

Application Note: The operational user guidance does not have to be contained in a single document. Guidance to users, administrators and application developers can be spread among documents or web pages. Rather than repeat information here, the developer should review the assurance activities for this component to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

Content and presentation elements:

AGD_OPE.1.2C

The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

Application Note: User and administrator are to be considered in the definition of user role.

AGD_OPE.1.3C

The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the OS in a secure manner.

AGD_OPE.1.4C

The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

Application Note: This portion of the operational user guidance should be presented in the form of a checklist that can be quickly executed by IT personnel (or end-users, when necessary) and suitable for use in compliance activities. When possible, this guidance is to be expressed in the eXtensible Configuration Checklist Description Format (XCCDF) to support security automation. Minimally, it should be presented in a structured format which includes a title for each configuration item, instructions for achieving the secure configuration, and any relevant rationale.

AGD_OPE.1.5C

The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.6C

The operational user guidance shall identify all possible modes of operation of the OS (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

AGD_OPE.1.7C

The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

AGD_OPE.1.8C

The operational user guidance shall be clear and reasonable.

Evaluator action elements:

AGD_OPE.1.9E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Evaluation Activities

AGD_OPE.1:
Some of the contents of the operational guidance are verified by the assurance activities in Section 5.1 Security Functional Requirements and evaluation of the OS according to the [CEM]. The following additional information is also required. If cryptographic functions are provided by the OS, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the OS. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the OS. The documentation must describe the process for verifying updates to the OS by verifying a digital signature – this may be done by the OS or the underlying platform. The evaluator will verify that this process includes the following steps: Instructions for obtaining the update itself. This should include instructions for making the update accessible to the OS (e.g., placement in a specific directory). Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature. The OS will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

AGD_PRE.1 Preparative Procedures (AGD_PRE.1)

Developer action elements:

AGD_PRE.1.1D

The developer shall provide the OS, including its preparative procedures.

Application Note: As with the operational guidance, the developer should look to the assurance activities to determine the required content with respect to preparative procedures.

Content and presentation elements:

AGD_PRE.1.2C

The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered OS in accordance with the developer's delivery procedures.

AGD_PRE.1.3C

The preparative procedures shall describe all the steps necessary for secure installation of the OS and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.

Evaluator action elements:

AGD_PRE.1.4E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_PRE.1.5E

The evaluator shall apply the preparative procedures to confirm that the OS can be prepared securely for operation.

Evaluation Activities

AGD_PRE.1:
As indicated in the introduction above, there are significant expectations with respect to the documentation—especially when configuring the operational environment to support OS functional requirements. The evaluator shall check to ensure that the guidance provided for the OS adequately addresses all platforms claimed for the OS in the ST.

5.2.4 Class ALC: Life-cycle Support

At the assurance level provided for OSs conformant to this PP, life-cycle support is limited to end-user-visible aspects of the life-cycle, rather than an examination of the OS vendor's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it is a reflection on the information to be made available for evaluation at this assurance level.

ALC_CMC.1 Labeling of the TOE (ALC_CMC.1)

This component is targeted at identifying the OS such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user.

Developer action elements:

ALC_CMC.1.1D

The developer shall provide the OS and a reference for the OS.

Content and presentation elements:

ALC_CMC.1.2C

The OS shall be labeled with a unique reference.

Application Note: Unique reference information includes:

- OS Name
- OS Version
- OS Description
- Software Identification (SWID) tags, if available

Evaluator action elements:

ALC_CMC.1.3E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Evaluation Activities

ALC_CMC.1:
The evaluator will check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator will check the AGD guidance and OS samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the OS, the evaluator will examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

ALC_CMS.1 TOE CM Coverage (ALC_CMS.1)

Given the scope of the OS and its associated evaluation evidence requirements, this component's assurance activities are covered by the assurance activities listed for ALC_CMC.1.

Developer action elements:

ALC_CMS.1.1D

The developer shall provide a configuration list for the OS.

Content and presentation elements:

ALC_CMS.1.2C

The configuration list shall include the following: the OS itself; and the evaluation evidence required by the SARs.

ALC_CMS.1.3C

The configuration list shall uniquely identify the configuration items.

Evaluator action elements:

ALC_CMS.1.4E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Evaluation Activities

ALC_CMS.1:
The "evaluation evidence required by the SARs" in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the OS is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the assurance activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component. Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.
The evaluator will ensure that the developer has identified (in guidance documentation for

application developers concerning the targeted platform) one or more development environments appropriate for use in developing applications for the developer's platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environment(s) are invoked (e.g., compiler and linker flags). The evaluator will ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled. The evaluator will ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification.

ALC_TSU_EXT.1 Timely Security Updates

This component requires the OS developer, in conjunction with any other necessary parties, to provide information as to how the end-user devices are updated to address security issues in a timely manner. The documentation describes the process of providing updates to the public from the time a security flaw is reported/discovered, to the time an update is released. This description includes the parties involved (e.g., the developer, carriers(s)) and the steps that are performed (e.g., developer testing, carrier testing), including worst case time periods, before an update is made available to the public.

Developer action elements:

ALC_TSU_EXT.1.1D

The developer shall provide a description in the TSS of how timely security updates are made to the OS.

ALC_TSU_EXT.1.2D

The developer shall provide a description in the TSS of how users are notified when updates change security properties or the configuration of the product.

Content and presentation elements:

ALC_TSU_EXT.1.3C

The description shall include the process for creating and deploying security updates for the OS software.

ALC_TSU_EXT.1.4C

The description shall include the mechanisms publicly available for reporting security issues pertaining to the OS.

Note: The reporting mechanism could include web sites, email addresses, as well as a means to protect the sensitive nature of the report (e.g., public keys that could be used to encrypt the details of a proof-of-concept exploit).

Evaluator action elements:

ALC_TSU_EXT.1.5E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Evaluation Activities

ALC_TSU_EXT.1:
The evaluator will verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates. The evaluator will verify that this description addresses the entire application. The evaluator will also verify that, in addition to the OS developer's process, any third-party processes are also addressed in the description. The evaluator will also verify that each mechanism for deployment of security updates is described.
The evaluator will verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the OS patching this vulnerability, to include any third-party or carrier delays in deployment. The evaluator will verify that this time is expressed in a number or range of days.
The evaluator will verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the OS. The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website.

5.2.5 Class ATE: Tests

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through the ATE_IND family, while the latter is through the AVA_VAN family. At the assurance level specified in this PP, testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

ATE_IND.1 Independent Testing – Conformance (ATE_IND.1)

Testing is performed to confirm the functionality described in the TSS as well as the administrative (including configuration and operational) documentation provided. The focus of the testing is to confirm that the requirements specified in Section 5.1 Security Functional Requirements being met, although some additional testing is specified for SARs in Section 5.2 Security Assurance Requirements. The Assurance Activities identify the additional testing activities associated with these components. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/OS combinations that are claiming conformance to this PP. Given the scope of the OS and its associated evaluation evidence requirements, this component's assurance activities are covered by the assurance activities listed for ALC_CMC.1.

Developer action elements:

ATE_IND.1.1D

The developer shall provide the OS for testing.

Content and presentation elements:

ATE_IND.1.2C

The OS shall be suitable for testing.

Evaluator action elements:

ATE_IND.1.3E

The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.1.4E

The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

Application Note: The evaluator will test the OS on the most current fully patched version of the platform.

Evaluation Activities

ATE_IND.1:
The evaluator will prepare a test plan and report documenting the testing aspects of the system, including any application crashes during testing. The evaluator shall determine the root cause of any application crashes and include that information in the report. The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's Assurance Activities.
While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered. The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary. The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the OS and its platform. This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS). The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results.
The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

5.2.6 Class AVA: Vulnerability Assessment

For the first generation of this protection profile, the evaluation lab is expected to survey open sources to discover what vulnerabilities have been discovered in these types of products. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. Until penetration tools are created and uniformly distributed to the evaluation labs, the evaluator will not be expected to test for these vulnerabilities in the OS. The labs will be expected to comment on the likelihood of these vulnerabilities given the documentation provided by the vendor. This information will be used in the development of penetration testing tools and for the development of future protection profiles.

AVA_VAN.1 Vulnerability Survey (AVA_VAN.1)

Developer action elements:

AVA_VAN.1.1D

The developer shall provide the OS for testing.

Content and presentation elements:

AVA_VAN.1.2C

The OS shall be suitable for testing.

Evaluator action elements:

AVA_VAN.1.3E

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.1.4E

The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the OS.

Application Note: Public domain sources include the Common Vulnerabilities and Exposures (CVE) dictionary for publicly-known vulnerabilities. Public domain sources also include sites which provide free checking of files for viruses.

AVA_VAN.1.5E

The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the OS is resistant to attacks performed by an attacker possessing Basic attack potential.

Evaluation Activities

AVA_VAN.1:
The evaluator will generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to find vulnerabilities that have been found in similar applications with a particular focus on network protocols the application uses and document formats it parses. The evaluator documents the sources consulted and the vulnerabilities found in the report.
For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

Appendix A - Implementation-Dependent Requirements

Implementation-Dependent Requirements are dependent on the TOE implementing a particular function. If the TOE fulfills any of these requirements, the vendor must either add the related SFR or disable the functionality for the evaluated configuration.

A.1 Widget Thing

This is a super description of this certain feature.

If this is implemented by the TOE, the following requirements must be included in the ST:

Appendix B - Inherently Satisfied Requirements

This appendix lists requirements that should be considered satisfied by products successfully evaluated against this Protection Profile. However, these requirements are not featured explicitly as SFRs and should not be included in the ST. They are not included as standalone SFRs because it would increase the time, cost, and complexity of evaluation. This approach is permitted by [CC] Part 1, **8.2 Dependencies between components**.

This information benefits systems engineering activities which call for inclusion of particular security controls. Evaluation against the Protection Profile provides evidence that these controls are present and have been evaluated.

Requirement	Rationale for Satisfaction
FIA_UAU.1 - Timing of authentication	FIA_AFL.1 implicitly requires that the OS perform all necessary actions, including those on behalf of the user who has not been authenticated, in order to authenticate; therefore it is duplicative to include these actions as a separate assignment and test.
FIA_UID.1 - Timing of identification	FIA_AFL.1 implicitly requires that the OS perform all necessary actions, including those on behalf of the user who has not been identified, in order to authenticate; therefore it is duplicative to include these actions as a separate assignment and test.
FMT_SMR.1 - Security roles	FMT_MOF_EXT.1 specifies role-based management functions that implicitly defines user and privileged accounts; therefore, it is duplicative to include separate role requirements.
FPT_STM.1 - Reliable time stamps	FAU_GEN.1.2 explicitly requires that the OS associate timestamps with audit records; therefore it is duplicative to include a separate timestamp requirement.
FTA_SSL.1 - TSF-initiated session locking	FMT_MOF_EXT.1 defines requirements for managing session locking; therefore, it is duplicative to include a separate session locking requirement.
FTA_SSL.2 - User-initiated locking	FMT_MOF_EXT.1 defines requirements for user-initiated session locking; therefore, it is duplicative to include a separate session locking requirement.
FAU_STG.1 - Protected audit trail storage	FPT_ACF_EXT.1 defines a requirement to protect audit logs; therefore, it is duplicative to include a separate protection of audit trail requirements.
FAU_GEN.2 - User identity association	FAU_GEN.1.2 explicitly requires that the OS record any user account associated with each event; therefore, it is duplicative to include a separate requirement to associate a user account with each event.
FAU_SAR.1 - Audit review	FPT_ACF_EXT.1.2 requires that audit logs (and other objects) are protected from reading by unprivileged users; therefore, it is duplicative to include a separate requirement to protect only the audit information.

Appendix C - Acronyms

Appendix D - Selection Rules

This rules in this appendix define which combinations of selections are considered valid. An ST is considered conforming only if it satisfies all rules.

Appendix E - Use Case Templates

E.1 Elephant-own device

Choose something other than:
  *
  *
Choose something other than:
  *

Appendix F - Acronyms

Acronym	Meaning
AES	Advanced Encryption Standard
API	Application Programming Interface
API	Application Programming Interface
ASLR	Address Space Layout Randomization
Base-PP	Base Protection Profile
CC	Common Criteria
CEM	Common Evaluation Methodology
CESG	Communications-Electronics Security Group
CMC	Certificate Management over CMS
CMS	Cryptographic Message Syntax
CN	Common Names
CRL	Certificate Revocation List
CSA	Computer Security Act
CSP	Critical Security Parameters
DAR	Data At Rest
DEP	Data Execution Prevention
DES	Data Encryption Standard
DHE	Diffie-Hellman Ephemeral
DNS	Domain Name System
DRBG	Deterministic Random Bit Generator
DSS	Digital Signature Standard
DSS	Digital Signature Standard
DT	Date/Time Vector
DTLS	Datagram Transport Layer Security
EAP	Extensible Authentication Protocol
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
EST	Enrollment over Secure Transport
FIPS	Federal Information Processing Standards
HMAC	Hash-based Message Authentication Code
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISO	International Organization for Standardization
IT	Information Technology
ITSEF	Information Technology Security Evaluation Facility
NIAP	National Information Assurance Partnership
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
OE	Operational Environment
OID	Object Identifier
OMB	Office of Management and Budget

OS	Operating System
PII	Personally Identifiable Information
PKI	Public Key Infrastructure
PP	Protection Profile
PP	Protection Profile
PP-Configuration	Protection Profile Configuration
PP-Module	Protection Profile Module
RBG	Random Bit Generator
RFC	Request for Comment
RNG	Random Number Generator
RNGVS	Random Number Generator Validation System
S/MIME	Secure/Multi-purpose Internet Mail Extensions
SAN	Subject Alternative Name
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SIP	Session Initiation Protocol
ST	Security Target
SWID	Software Identification
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
TSS	TOE Summary Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USB	Universal Serial Bus
VM	Virtual Machine
XCCDF	eXtensible Configuration Checklist Description Format
XOR	Exclusive Or
app	Application

Appendix G - Bibliography

Identifier	Title
[CC]	Common Criteria for Information Technology Security Evaluation - • Part 1: Introduction and General Model, CCMB-2017-04-001, Version 3.1 Revision 5, April 2017. • Part 2: Security Functional Components, CCMB-2017-04-002, Version 3.1 Revision 5, April 2017. • Part 3: Security Assurance Components, CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.
[CEM]	Common Evaluation Methodology for Information Technology Security - Evaluation Methodology, CCMB-2012-09-004, Version 3.1, Revision 4, September 2012.
[CESG]	CESG - End User Devices Security and Configuration Guidance
[CSA]	Computer Security Act of 1987, H.R. 145, June 11, 1987.
[OMB]	Reporting Incidents Involving Personally Identifiable Information and Incorporating the Cost for Security in Agency Information Technology Investments, OMB M-06-19, July 12, 2006.