collaborative Protection Profile Module for Full Drive Encryption – Enterprise Management

March 23rd, 2018

Version 2.0

Acknowledgements

This collaborative Protection Profile Module (cPP-Module) was developed by the Full Drive Encryption international Technical Community with representatives from industry, Government agencies, Common Criteria Test Laboratories, and members of academia.

0. Preface

0.1 Objectives of Document

This document presents the Common Criteria (CC) collaborative Protection Profile Module (cPP-Module) to express the security functional requirements (SFRs) and security assurance requirements (SARs) for an Enterprise Management capability for Full Drive Encryption. The Evaluation Activities that specify the actions the evaluator performs to determine whether a product satisfies the SFRs captured within this cPP-Module are described in *Supporting Document (Mandatory Technical Document) Full Drive Encryption: Enterprise Management September 2015.*

A complete FDE solution requires both an Authorization Acquisition (AA) component and Encryption Engine (EE) component. It may not require an Enterprise Management capability, which is why this capability is expressed as a cPP-Module that may optionally extend a TOE that conforms to the AA cPP or both the AA and EE cPPs.

0.2 Scope of Document

The scope of the cPP-Module within the development and evaluation process is described in the Common Criteria for Information Technology Security Evaluation, Revision 5. In particular, a cPP defines the IT security requirements of a technology specific type of TOE and specifies the functional and assurance security requirements to be met by a compliant TOE. A cPP-Module then extends these requirements by defining a uniquely-identified set of capabilities that can be used to optionally extend the security claims made by a product that conforms to a “base” cPP.

0.3 Intended Readership

The target audiences of this cPP-Module are developers, CC consumers, system integrators, evaluators and schemes.

0.4 Related Documents

Protection Profiles

[FDE – AA] collaborative Protection Profile for Full Drive Encryption – Authorization Acquisition, Version 2.0, May 18, 2017

[FDE – EE] collaborative Protection Profile for Full Drive Encryption – Encryption Engine, Version 2.0, May 18, 2017

1 Common Criteria[1]

[CC1] Common Criteria for Information Technology Security Evaluation,
Part 1: Introduction and General Model,
CCMB-2017-04-001, Version 3.1 Revision 5, April 2017.

[CC2] Common Criteria for Information Technology Security Evaluation,
Part 2: Security Functional Components,
CCMB-2017-04-002, Version 3.1 Revision 5, April 2017.

[CC3] Common Criteria for Information Technology Security Evaluation,
Part 3: Security Assurance Components,
CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.

[CEM] Common Methodology for Information Technology Security Evaluation,
Evaluation Methodology,
CCMB-2017-04-004, Version 3.1, Revision 5, April 2017.

[SD] Supporting Document (Mandatory Technical Document), Full Drive Encryption: Enterprise Management January 2018.

2

[1] For details see http://www.commoncriteriaportal.org/

0.5 Revision History

Version	Date	Description
2.0	March 23rd, 2018	Draft published for public review (version set to 2.0 for consistency with AA and EE PPs)

Contents

Figures / Tables

1. PP Introduction

1.1 PP Reference Identification

PP Reference: collaborative Protection Profile Module for Full Drive Encryption – Enterprise Management

PP Version: 2.0

PP Date: March 23rd, 2018

1.2 Introduction to the FDE Collaborative Protection Profiles (cPPs) Effort

The purpose of the first set of Collaborative Protection Profiles (cPPs) for *Full Drive Encryption (FDE): Authorization Acquisition (AA)* and *Encryption Engine (EE)* is to provide requirements for Data-at-Rest protection for a lost device that contains data. These cPPs allow FDE solutions based in software and/or hardware to meet the requirements. For more information on the *Authorization Acquisition (AA)* and *Encryption Engine (EE)*, please see the front matter of the relevant cPP.

The purpose of the *Enterprise Management (EM) Module* is to provide security critical requirements for Enterprise Management software that is used to manage systems in an enterprise that contain FDE solutions. Such software is used to provision and administer such solutions and maintain backup means of authorizing the systems, should a primary authorization be lost or forgotten.

The *Enterprise Management Module* builds on top of the *FDE cPP – Authorization Acquisition* and details the security requirements and assurance activities necessary for the common Enterprise features that that the iTC tackled in Version 1 (see Figure 1). An endpoint which is centrally managed by an IT organization presents unique new challenges for a data-at-rest encryption solution. This addition to the *FDE cPP – Authorization Acquisition* addresses the following scenarios over and above what was addressed in the first release of the cPP:

- Managing the DEK, KEK and encryption policy from a Management Server
- Providing for multi-user access to an endpoint protected by a compliant FDE solution
- Providing for remote authentication of the user (Figure 3)
- Providing for user recovery scenarios when a user's credential is lost or forgotten.

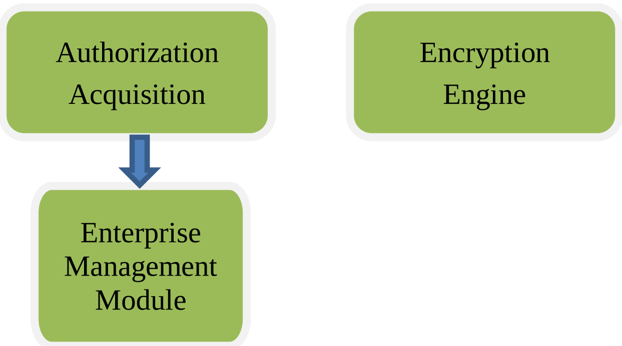


Figure 1: FDE Components

This TOE description defines the scope and functionality of the *Enterprise Management Module*, and the Security Problem Definition describes the assumptions made about the operating environment and the threats to the Enterprise Management Module that the cPP requirements address.

1.3 Implementations

The *Enterprise Management Module* solutions vary with implementation and vendor combinations. The *Enterprise Management Module* is an extension to the FDE AA cPP. Therefore, it is assumed that one vendor will be bringing in one TOE for evaluation. When a customer acquires an Enterprise Managed FDE solution, they will either obtain a single vendor product that meets the AA + EE set of Base-PPs + EM Module, or two products, one which meets the AA cPP + EM Module and one which meets the EE cPP.

It should be noted that in the case that a management engine is used to interface with EE, it is assumed there is at least a minimal AA that provides an interface between the two.

The table below illustrates a few *examples* for certification.

Table 1: Examples of cPP Implementations

Implementation	cPP	Description
Host + EM	AA + EM Module	Host software provides the interface to a self-encrypting drive and Administrative software that allows enterprise management of the interface.
Software FDE	AA + EM Module + EE	An enterprise manageable software full drive encryption solution
Hybrid	AA + EM Module + EE	A single vendor's combination of hardware (e.g. hardware encryption engine or cryptographic co-processor) and enterprise manageable software

1.4 Target of Evaluation (TOE) Overview

The target of evaluation for this cPP-Module is the Enterprise Management (EM) function of an FDE. The EM function is designed to augment the claims made in the FDE AA cPP; therefore, this functionality is intended to be evaluated in conjunction with a TOE that also claims conformance to this cPP at minimum.

The following sections provide an overview of the security functionality of this PP-module.

1.4.1 Enterprise Management Introduction

The Enterprise Management Module objectives focus on access recovery and policy enforcement. The optional EM is responsible for maintaining a mechanism for recovering access to the EE by the following interactions with the AA:

- Verification of authority to utilize the recovery mechanism and AA requesting credentials
- Recovery of credentials
- Securely providing credentials to the AA

The AA then uses the credentials to produce a Border Encryption Value (BEV) for a different key chain than normally used by the user, to provide access (via the EE) to the encrypted data

The EM is responsible for allowing or denying a requested action based on satisfying access requirements of a back end server (e.g. Active Directory or a different LDAP). The EM may provide support for multiple users being able to request the action.

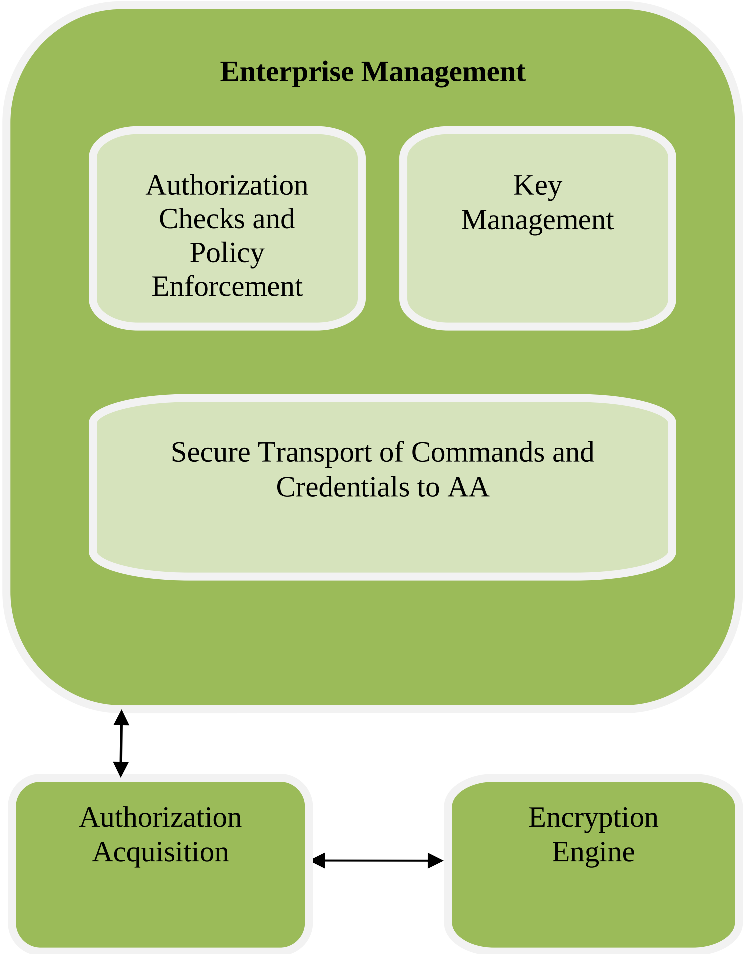


Figure 2: Enterprise Management Details

Figure 2 illustrates the components within Enterprise Management and its relationship with AA.

1.4.2 Enterprise Management Security Capabilities

The *Enterprise Management (EM) Module* is responsible for maintaining the ability of the AA to authorize the Encryption Engine to perform its action, in the event that the normal user is not able to. This may include stituations where the user has lost or forgotten credentials necessary to authenticate to the AA, the user is no longer employeed by the enterprise, or may include include situations where the user has lost control of the physical device, and cryptographic wiping of the data on the device is being requested through key sanitization.

The EM interfaces with the AA to provide these facilities. It is responsible for verifying authorization of the requestor's authority to perform an operation before releasing key material either to the requestor or to the AA on the requestor's behalf. That key material may include a

BEV which the AA uses as part of a key chain used by the AA and EE, but the Enterprise Management Module itself does not interface with the EE directly. It is responsible for maintaining the security of any key material it stores. Since differing users have differing access rights, it is also the responsibility of the EM to make certain that recovery material cannot be used by an AA to perform authorization of actions other than those authorized by the authenticated authorizing party.

The Enterprise Management Module uses approved cryptography to generate, handle, and protect key materials so as to force an adversary who obtains an unpowered lost or stolen platform without the authorization factors or intermediate keys to exhaust the encryption key space of intermediate keys or DEK to obtain the data.

1.4.3 The TOE and the Operational/Pre-Boot Environments

The environment in which the EM functions is expected to exist is on a back end server, not on the system that contains the EE. It is expected to have secure access to a certified LDAP (e.g. Active Directory) and access to a certified means of storing key material when not in use. The EM shall not have the ability to access the secured stored key material without verification of access authority by the LDAP.

The Operating System environment may make a full range of services available to the Enterprise Management Module, including hardware drivers, cryptographic libraries, and perhaps other services external to the TOE (see Figure 3).

The EM TOE may include or leverage features and functions within the operational environment.

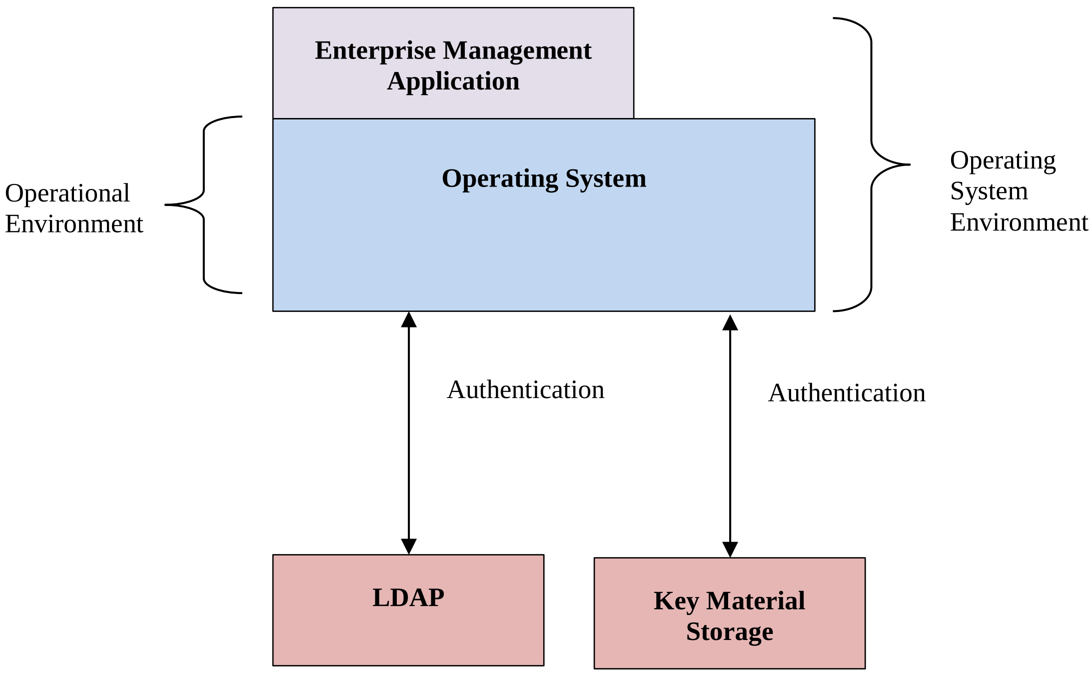


Figure 3: Operational Environment

1.5 TOE Use Case

The use case for a product conforming to the FDE cPPs is to protect data at rest on a device that is lost or stolen while powered off without any prior access by an adversary. The use case where an adversary obtains a device that is in a powered state and is able to make modifications to the environment or the TOE itself (e.g., evil maid attacks) is not addressed by these cPPs (i.e., FDE - AA and FDE - EE).

While that use case is still true for the *Enterprise Management Module*, this PP-module also expands the use case to include protecting the communications between the Enterprise Management Server and the client device through the use of a trusted channel. It also expands the use case to include the optional abilities of the EM to interact with the AA (with proper authorization) to direct it to perform sanitation of keys and material on the device or to issue a recovery credential to reset the authentication factor if it has been lost.

2. CC Conformance

As defined by the references [CC1], [CC2] and [CC3], this cPP-Module conforms to the requirements of Common Criteria v3.1, Revision 5. This cPP-Module is conformant to CC v3.1, r5, CC Part 2 extended, and CC Part 3 conformant. Extended component definitions can be found in Appendix C.

The methodology applied for the cPP-Module evaluation is defined in [CEM].

This cPP-Module satisfies the following Assurance Families: ACE_INT.1, ACE_CCL.1, ACE_SPD.1, ACE_ECD.1, ACE_OBJ.1, ACE_REQ.1, ACE_MCO.1, ACE_CCO.1, APE_CCL.1, APE_ECD.1, APE_INT.1, APE_OBJ.1, APE_REQ.1, and APE_SPD.1.

This cPP-Module does not claim conformance to another cPP.

In order to be conformant to this cPP-Module, a TOE must demonstrate *Exact Conformance*. *Exact Conformance* is defined as the ST containing all of the requirements in section 5 of this cPP-Module, and potentially requirements from Appendix A or Appendix B of this cPP-Module. While iteration is allowed, no additional requirements (from CC parts 2 or 3) are allowed to be included in the ST, except those belonging to any other Protection Profiles claimed by the TOE (e.g. FDE – AA). Further, no requirements in section 5 of this cPP-Module are allowed to be omitted.

2.1 Components Statement

The following PP-Configurations that include this cPP-Module are permitted:

- [FDE – AA] (Base-PP) and this cPP-Module
- [FDE – AA] and [FDE – EE] (set of Base-PPs), and this cPP-Module

2.2 Consistency Rationale

2.2.1 AA as Base-PP

The TOE type for both [FDE – AA] and this cPP-Module is Full Drive Encryption. The security functionality described in [FDE – AA] relates to the method by which an FDE TOE collects one or more authorization factors to generate a BEV for an encryption engine. A TOE that includes an Enterprise Management (EM) capability can include this cPP-Module if it is deployed in an environment where a centralized management server can be used to configure multiple AA instances over a network. The threats defined for this cPP-Module represent attack scenarios that are unique to an environment where TSF data is traversing a network from a centralized management server to one or more AA instances. The TOE security objectives and related SFRs have been written to mitigate these threats and to satisfy those security functions from the AA that the EM capability includes to allow for distributed execution of these functions.

Some threats defined in this cPP-Module are augmentations of threats that already exist in the Base-PP but can be exploited in a different way when the EM capability is present. These threats are identified in section 3 using the same name as the original threat defined in the Base-

PP followed by a slash (/) and a secondary name that qualifies the specific nature of the modified threat.

2.2.2 AA and EE as set of Base-PPs

The consistency with the cPP-Module and the Base-PP when both [FDE – AA] and [FDE – EE] are claimed as a set of Base-PPs is similar to the case where only [FDE – AA] is the Base-PP. In particular, a TOE that includes both AA and EE capabilities will validate the BEV and perform drive encryption (unlike in the case where just [FDE – AA] is the Base-PP) but all of the functions provided by the EM capability are used to interface with the AA functionality of the FDE. Therefore, the inclusion of [FDE – EE] within the TOE boundary is non-interfering with respect to the security functionality described in this cPP-Module.

Some threats defined in this cPP-Module are augmentations of threats that already exist in one of the set of Base-PPs but can be exploited in a different way when the EM capability is present. These threats are identified in section 3 using the same name as the original threat defined in the Base-PP followed by a slash (/) and a secondary name that qualifies the specific nature of the modified threat.

Note that even in the case where both the AA and EE are claimed as a set of Base-PPs, this cPP-Module will use the term “Base-PP” to refer to [FDE – AA] since all functionality described in the cPP-Module interfaces with the AA component of the TOE.

3. Security Problem Definition

3.1 Threats

This section provides a narrative that describes how the requirements mitigate the mapped threats. A requirement may mitigate aspects of multiple threats. A requirement may only mitigate a threat in a limited way. Some requirements are optional, either because the TSF fully mitigates the threat without the additional requirement(s) being claimed or because the TSF relies on its Operational Environment to provide the functionality that is described by the optional requirement(s).

A threat consists of a threat agent, an asset and an adverse action of that threat agent on that asset. The threat agents are the entities that put the assets at risk if an adversary obtains a lost or stolen storage device. Threats drive the functional requirements for the target of evaluation (TOE).

For instance, one threat below is T.UNAUTHORIZED_DATA_ACCESS/SERVER. The threat agent is a malicious actor that is attempting to access the Management Server component of the TOE (i.e. the component defined by this cPP-Module). The asset is the data on the Management Server, while the adverse action is to attempt to obtain data from the Management Server which could lead to the compromise of one or more drives that are managed by the TSF. This threat drives the functional requirements for the encrypted storage device (TOE) to authorize who can use the TOE to access the data used to interact with one or more encrypted drives. Since possession of the KEK, DEK, intermediate keys, authorization factors, submasks, and random numbers or any other values that contribute to the creation of keys or authorization factors could allow an unauthorized user to defeat the encryption, this SPD considers keying material equivalent to the data in importance and they appear among the other assets addressed below.

It is important to reemphasize at this point that this cPP-Module does not expect the product (TOE) to defend against a malicious agent that has unrestricted logical access to the system on which the Management Server resides. Security of the TOE, which includes the drive(s) to be protected as well as the capability for the drive(s) to be managed, requires appropriate physical and logical protections as part of a defense-in-depth strategy.

(T. UNAUTHORIZED_DATA_ACCESS) is an extension from the threat in [FDE – AA] to incorporate the threat of an attacker accessing the data on the encrypted drive by getting access to a protected drive, attaching it to a host system controlled by the attacker and using the key material, BEV, or optionally a recovery credential to access the data. The Base-PP addresses the primary threat of unauthorized disclosure of recovery material protected by the drive(s); this adds attribution of the key material to the drive.

[Mandatory SFRs: FPT_KYP_EXT.3;

Optional SFRs: None]

Rationale: FPT_KYP_EXT.3 requires that the key material, BEV, and optionally recovery credentials be uniquely associated with the encrypted drive at a minimum. Additionally, key material may also be associated with a specific system or user to prevent an attacker from accessing the data on the encrypted drive by inserting the drive in a host with weaker security. A product which distributes keys to meet the requirements of FPT_KYP_EXT.3 will additionally prevent an attacker from gaining access to the encrypted data.

(T.UNAUTHORIZED_DATA_ACCESS/SERVER) is an extension from the threat in [FDE – AA] to incorporate the threat of an attacker accessing the Management Server. The Base-PP addresses the primary threat of unauthorized disclosure of recovery material protected by the drive(s); this adds the Management Server to the scope of the threat.

[Mandatory SFRs: FIA_UAU.1, FIA_UID.1, FMT_MTD.1, FMT_SMR.2;

Optional SFRs: FCS_COP.1(f)/Server, FCS_VAL_EXT.2, FIA_X509_EXT.1/Server, FIA_X509_EXT.2/Server]

Rationale: FIA_UAU.1 requires the administrator to be authenticated prior to allowing the administrator to manage the product via the remote console. FIA_UID.1 requires the admin to be identified prior to allowing the administrator to manage the product via the remote console. FMT_MTD.1 requires that actions which result in changes to key material, user authentication policy and recovery are constrained to administrators and specific times. FMT_SMR.2 requires users be assigned roles. FCS_VAL_EXT.2, if selected, requires user authentication to be validated by the Operational Environment or the TOE prior to releasing the BEV.

The Optional capability which may be provided by the TSF would include encryption of data stored on the server, as validated by FCS_COP.1(f)/Server; and certificate-based authentication, validated by FIA_X509_EXT.2/Server and validation, as validated by FIA_X509_EXT.1/Server.

(T.KEYING_MATERIAL_COMPROMISE/SERVER) – Possession of any of the keys, authorization factors, submasks, and random numbers or any other values that contribute to the creation of keys or authorization factors could allow an unauthorized user to defeat the encryption. This cPP-Module considers possession of key material of equal importance to the data itself. Threat agents may look for key material in unencrypted storage on the Management Server and in external databases in the operating environment (OE), e.g. SQL database.

[Mandatory SFRs: FCS_AFA_EXT.1, FCS_KYC_EXT.1/Server, FPT_KYP_EXT.1, FPT_KYP_EXT.2, FPT_KYP_EXT.3, FCS_SMC_EXT.1/Server, FMT_SMF.1/Server, FPT_ITT.1;

Optional SFRs: FCS_CKM.1(a)/Server, FCS_VAL_EXT.2, FCS_CKM.4(a)/Server, FCS_RBG_EXT.1/Server, FCS_CKM.2/Server, FCS_CKM.2, FCS_COP.1(b)/Server, FCS_COP.1(c)/Server, FCS_COP.1(d)/Server, FCS_COP.1(g)/Server, FCS_KDF_EXT.1/Server, FCS_SNI_EXT.1/Server, FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FMT_MOF.1/Server]

Rationale: The keying material that threat agents may attempt to compromise are generated by the TOE as specified by FCS_CKM.1(a)/Server (or by the Operational Environment if this optional SFR is not claimed). One or more submasks

[FCS_AFA_EXT.1] may be chained [FCS_KYC_EXT.1/Server] to produce the BEV. The server key chain can be maintained by several methods, including:

- Key generation [FCS_CKM.1(a)/Server]
- Key establishment [FCS_CKM.2/Server]
- Key distribution [FCS_CKM.2]
- Key derivation [FCS_KDF_EXT.1/Server]
- Key attribution [FPT_KYP_EXT.3]
- Key combining [FCS_COP.1(b)/Server]
- Key derivation [FCS_COP.1(c)/Server]
- Key wrapping [FCS_COP.1(d)/Server]
- Key transport [FCS_COP.1(e)/Server]
- Key combining [FCS_SMC_EXT.1/Server]
- Key storage [FPT_KYP_EXT.1, FPT_KYP_EXT.2]
- Key encryption [FCS_COP.1(g)]
- Salt, Nonce, and IV generation [FCS_SNI_EXT.1]

Key chains may be maintained using asymmetric [FCS_CKM.1(a)/Server] and/or symmetric [FCS_CKM.1(b)/Server].

These requirements ensure the BEV is properly generated and protected. If selected, FMT_MOF.1/Server ensures that only administrators can select the encryption algorithms and key sizes. Only administrators can perform management functions on the Enterprise Management Server as defined in FMT_SMF.1/Server.

FCS_KYC_EXT.1/Server extends the requirements of [FDE - AA] key chaining to key chains generated or maintained by the Server.

FPT_ITT.1 ensures that keys and key material transported between the EM and the AA are protected from disclosure, modification, deletion, substitution, reordering or insertion.

FPT_KYP_EXT.1 ensures unwrapped key material is not stored in non-volatile memory minimizing the exposure of plaintext keys and key material.

The following optional components ensure that key material is not exposed through the communication channel between an Enterprise Server and the AA, if remote management is supported by the TSF. The requirements for establishing keys are validated by FCS_CKM.2/Server which relies on one or more of the following SFR's to implement secure communications:

- FCS_HTTPS_EXT.1,
- FCS_IPSEC_EXT.1,
- FCS_SSHS_EXT.1,
- FCS_SSHC_EXT.1,
- FCS_TLSC_EXT.1 (and optionally FCS_TLSC_EXT.3 depending on the claimed ciphersuites), and
- FCS_TLSS_EXT.1 (and optionally FCS_TLSS_EXT.3 depending on the claimed ciphersuites).

The various iterations of FCS_COP.1/Server as well as FCS_RBG_EXT.1/Server all validate that the cryptography used to initiate and protect the communication channel protocols between the Enterprise Server and the AA, if remote management is supported by the TSF. If implemented on the server, FCS_CKM.4(a)/Server ensures proper destruction of keys and key material on the server when no longer needed.

In order to ensure that a BEV is only released to the appropriate endpoint, FCS_KYP_EXT.3 ensures that there is attribution of the endpoint or encrypted disk and the BEV. The optional Server requirement FCS_CKM.2 ensures that if the BEV is communicated between the server and the endpoint, keys distributed by the server are given to the correct endpoint for the purpose of delivering the BEV.

(T.MAN_IN_THE_MIDDLE) The cPP-Module addresses the threat of an attacker listening on the intra-TOE communication between the Management Server and the AA to obtain the user's credential, keys, or recovery material.

[Mandatory SFRs: FPT_ITT.1;

Optional SFRs: FCS_CKM.1(a)/Server, FCS_COP.1(a)/Server, FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.3, FCS_TLSS_EXT.1, FCS_TLSS_EXT.3]

Rationale: FPT_ITT.1 ensures protection of intra TOE communication from disclosure, modification, reordering, substitution, or deletion If server side key generation is implemented, FCS_CKM.1(a)/Server ensures sufficiently strong keys correctly generated on the server to meet the requirements of FTP_TRP.1. Products implementing cryptographic communication protocols between the server and managed endpoints must meet the requirements for the specific protocols as defined in any of {FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1}. If TLS is supported, then FCS_TLSC_EXT.3 and/or FCS_TLSS_EXT.3 may also apply, depending on the claimed TLS ciphersuites. If the EM Server generates signatures to request or verify certificates, FCS_COP.1(a)/Server ensures correct cryptographic operation in signature generation process.

(T.UNAUTHORIZED_ADMINISTRATOR_ACCESS) The cPP-Module addresses the threat of an attacker masquerading as an administrator to the Management Server.

[Mandatory SFRs: FIA_UAU.1, FIA_UID.1;

Optional SFRs: None]

Rationale: FIA_UAU.1 requires that the administrator be authenticated by the EM. The administrator is required by FIA_UID.1 to successfully authenticate to the EM prior to being permitted to perform management functions on behalf of the administrator.

(T.UNTRUSTED_COMMUNICATION_CHANNELS) The cPP-Module address the threat of an attacker targeting the Management Server using insecure tunneling protocols or the presence of an unencrypted path to disclose keys, key material, or recovery material transferred between the endpoint and the Management Server.

[Mandatory SFRs: FTP_TRP.1;

Optional SFRs: FCS_COP.1(a)/Server, FCS_PCC_EXT.1/Server, FCS_RBG_EXT.1/Server, FIA_X509_EXT.1/Server, FIA_X509_EXT.2/Server, FIA_X509_EXT.3/Server, FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSS_EXT.1, FCS_TLSS_EXT.3]

Rationale: FPT_TRP.1 addresses the threat of disclosure of keys, key material, or recovery material transferred between the endpoint or a remote administrator and the Management Server when transmitted over untrusted communication channels by requiring use of IPsec, SSH, TLS, and/or TLS/HTTPS protocols when such data passes through those channels.

The selection-based communication protocol SFR's FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, and FCS_TLSC_EXT.1 ensure correct implementation of the protocols required by FTP_TRP.1. If TLS is supported, then FCS_TLSS_EXT.3 may also apply, depending on the claimed TLS ciphersuites. FCS_RBG_EXT.1/Server ensures sufficiently strong keys are generated for the communication protocols previously referenced. FIA_X509_EXT.1/Server, FIA_X509_EXT.2/Server, and FIA_X509_EXT.3/Server ensure the communication channel is established only with a server that is authenticated. FCS_COP.1(a)/Server ensures correct generation of cryptographic signatures.

If the TSF generates password authorization factors, the requirements of FCS_PCC_EXT.1/Server ensure that the password data is not subjected to unauthorized disclosure or brute force attack.

3.2 Assumptions

Assumptions about the TOE's Operational Environment that must remain true in order to mitigate the threats defined in section 3.1 appear below. Note that these assumptions supplement those that exist in the Base-PP; both sets of assumptions are expected to be satisfied by a conformant ST.

(A.NON-MALICIOUS_ADMIN) Administrators are assumed to be non-malicious, competent, and correctly trained.

(A.SECURED_CONFIGURATION) The Management Server and the remote endpoints are assumed to be installed and configured in accordance with their evaluated configuration.

[OE.SECURED_CONFIGURATION]

(A.SECURED_ENVIRONMENT) Any environmental components required to support the functionality of the Management Server (e.g. underlying operating system, firewall, database) are assumed to be installed and configured in accordance with its evaluated configuration.

[OE.SECURED_ENVIRONMENT]

(A.PHYSICAL/SERVER) This assumption extends the A.PHYSICAL assumption in the Base-PP to assume that the platform on which the Management Server resides is assumed to be physically protected in its Operational Environment and not subject to physical attacks that compromise the security and/or interfere with the platform's correct operation.

[OE.PHYSICAL/SERVER]

(A.ENVIRONMENTAL_STORAGE) Any key storage mechanism provided by the Operational Environment is able to provide the same level of security as a TOE-internal storage mechanism that is conformant to this PP-Configuration.

[OE.ENVIRONMENTAL_STORAGE]

3.3 Organizational Security Policies

In order to provide an appropriate level of security, the organization is expected to adhere to the following organizational security policies in order to satisfy the security objectives for the Operational Environment.

There are no organizational security policies that are mandatory for this cPP-Module. Note however that in the case where recovery credentials are supported, the organization is expected to implement a policy that ensures sufficiently strong recovery credentials are used to mitigate the use of the recovery credential as an attack vector. Refer to Appendix B.1 for details.

4. Security Objectives

4.1 Security Objectives for the Operational Environment

The Operational Environment of the TOE implements technical and procedural measures to assist the TOE in correctly providing its security functionality. This part wise solution forms the security objectives for the Operational Environment and consists of a set of statements describing the goals that the Operational Environment should achieve.

Note that these objectives supplement those that exist in the Base-PP; both sets of objectives are expected to be satisfied by the Operational Environment defined in a conformant ST.

(OE.SECURED_CONFIGURATION) The Management Server and remote endpoints are configured in accordance with its associated operational guidance so that the level of security that is provided by the TOE is consistent with its evaluated configuration.

> Rationale: The TSF may provide security mechanisms that require configuration to be performed after it has been installed. A trusted administrator will satisfy this objective by configuring the TOE in accordance with its operational guidance. The AA component of the TOE (i.e. one or more software instances that conforms to the Base-PP) may require environmental configuration prior to secure use.

(OE.SECURED_ENVIRONMENT) The components of the Management Server's underlying platform are configured in accordance with their associated operational guidance so that the TOE is deployed in an environment that is consistent with its evaluated configuration.

> Rationale: Administrators are trusted to follow the operational guidance that is provided for secure installation and configuration of the TOE, which includes any aspects of its underlying platform (such as an operating system, firewall, or database).

(OE.PHYSICAL/SERVER) The Operational Environment will provide a secure physical computing space such than an adversary is not able to make modifications to the environment or to the TOE itself, which includes the Management Server.

> Rationale: The expected deployment of the TOE is in an enterprise computing environment. The Management Server can reasonably be expected to be deployed in a secured environment because it does not provide functionality that would necessitate its deployment in a high-risk public-facing environment.

(OE.ENVIRONMENTAL_STORAGE) If the TOE relies on the Operational Environment for key storage, the storage mechanism will provide at least the same level of security as a TOE-internal storage mechanism that is conformant to this PP-Configuration.

> Rationale: The strength of the keys used by the TOE are limited by the strength of the key storage if it is computationally less difficult to disclose the key than to break it. Therefore, any environmental storage that the TSF relies on needs to ensure that it is at least as difficult to break as the keys themselves.

5. Security Functional Requirements

The individual security functional requirements are specified in the sections below. Based on selections made in these SFRs it will also be necessary to include some of the selection-based SFRs in Appendix B. Additional optional SFRs may also be adopted from those listed in Appendix A for those functions that may be provided by the TOE but are not strictly necessary.

The Evaluation Activities defined in [SD] describe actions that the evaluator will take to determine compliance of a particular TOE with the SFRs. The content of these Evaluation Activities will therefore provide more insight into deliverables required from TOE Developers.

5.1 Conventions

The conventions used in descriptions of the SFRs are as follows:

- Assignment: Indicated with *italicized text*;
- Refinement made by PP author: Indicated with **bold text** or ~~strikethroughs~~ for text that is added to or removed from the original SFR;
- Selection: Indicated with underlined text;
- Assignment within a Selection: Indicated with *italicized and underlined text*;
- Iteration: Indicated by appending the SFR with parentheses that contain a letter that is unique for each iteration, e.g. (a), (b), (c) and/or with a slash (/) followed by a descriptive string for the SFR’s purpose, e.g. /Server.

SFR text that is bold, italicized, and underlined indicates that the original SFR defined an assignment operation but the PP author completed that assignment by redefining it as a selection operation, which is also considered to be a refinement of the original SFR.

If the selection or assignment is to be completed by the ST author, it is preceded by ‘selection:’ or ‘assignment:’. If the selection or assignment has been completed by the PP author and the ST author does not have the ability to modify it, the proper formatting convention is applied but the preceding word is not included.

Extended SFRs (i.e. those SFRs that are not defined in CC Part 2) are identified by having a label ‘_EXT’ at the end of the SFR name.

5.2 SFR Architecture

The following table lists the SFRs that are mandated by this cPP-Module.

Table 2: TOE Security Functional Requirements

Functional Class	Functional Components
Cryptographic Support (FCS)	FCS_KYC_EXT.1/Server Key Chaining (Initiator) (Management Server)
	FCS_SMC_EXT.1/Server Submask Combining (Management Server)
Identification and Authentication (FIA)	FIA_UAU.1 Timing of Authentication
	FIA_UID.1 Timing of Identification
Security Management (FMT)	FMT_MTD.1 Management of TSF Data

	FMT_SMF.1/Server Specification of Management Functions (Management Server)
	FMT_SMR.2 Restrictions on Security Roles
Protection of the TSF (FPT)	FPT_ITT.1 Basic Internal TSF Data Transfer Protection
	FPT_KYP_EXT.2 Storage of Protected Key and Key Material
	FPT_KYP_EXT.3 Attribution of Protected Key and Key Material
Trusted Path/Channels (FTP)	FTP_TRP.1 Trusted Path

5.3 SFRs to be Modified from Base-PP

In order for the PP-Configuration to meet the threats defined in this cPP-Module, it is necessary to further refine some SFRs that are defined in the Base-PP. The modified SFRs listed below are to be substituted for their Base-PP counterparts by the ST author. [SD] defines the Evaluation Activities that are to be performed against the refined SFRs; these can be consulted by the evaluator when evaluating the modified SFRs in place of the Supporting Documents for the Base-PP.

5.3.1 Class: Cryptographic Support (FCS)

As a general note, the Base-PP includes a large number of optional SFRs related to the generation and protection of the BEV and key chain. Depending on the implementation, a TOE that conforms to this PP-Configuration may have these functions performed by some combination of the AA, the Management Server, and the Operational Environment. The ST author shall include all optional SFRs from the Base-PP if they are satisfied by the AA and/or the Management Server and clearly indicate in the TSS and KMD which component of the TOE is responsible for performing these functions.

FCS_AFA_EXT.1 Authorization Factor Acquisition

FCS_AFA_EXT.1.1 Refinement: The TSF shall accept the following authorization factors: [selection:

- a submask derived from a password authorization factor conditioned as defined in FCS_PCC_EXT.1,
- an external Smartcard factor that is at least the same bit-length as the DEK, and is protecting a submask that is [selection: generated by the TOE (using the RBG as specified in FCS_RBG_EXT.1), generated by the Host Platform] protected using RSA with key size of [selection: 2048 bits, 3072 bits, 4096 bits], with user presence proved by presentation of the smartcard and [selection: none, an OE defined PIN, a configurable PIN].
- an external Smartcard factor that is at least the same bit-length as the DEK, and is protecting a submask that is generated by the Host Platform, protected using RSA (key size of 2048 or above),
- an external USB token factor that is at least the same security strength as the BEV, and is providing a submask generated by the TOE, using the RBG as specified in FCS_RBG_EXT.1,

- an external USB token factor that is at least the same security strength as the BEV, and is providing a submask generated by the Host Platform,

- **a recovery credential generated by the TOE and conditioned as defined in FCS_PCC_EXT.1**

].

Application Note: *This SFR was modified from its original definition in the Base-PP to allow for the possible selection of a recovery credential as an authorization factor.*

This requirement specifies what authorization factors the TOE accepts from the user. A password entered by the user is one authorization factor that the TOE must be able to condition, as specified in FCS_PCC_EXT.1 from the AA cPP. Another option is a smart card authorization factor, with the differentiating feature being how the value is generated – either by the TOE's RBG or by the platform. An external USB token may also be used, with the submask value generated either by the TOE's RBG or by the platform. If a user-created recovery password is accepted by the TOE, the TOE must be able to condition, as specified in FCS_PCC_EXT.1.

The TOE may accept any number of authorization factors, and these are categorized as "submasks". The ST author selects the authorization factors they support, and there may be multiple methods for a selection.

Use of multiple authorization factors is preferable; if more than one authorization factor is used, the submasks produced must be combined using FCS_SMC_EXT.1.

5.3.2 Class: Protection of the TSF (FPT)

FPT_KYP_EXT.1 Protection of Key and Key Material

This SFR is not modified from the Base-PP. Note however that in the PP-Configuration it also applies to the Management Server. If the TSF provides different methods of key and key material protection for each individual component of the TOE, the ST author shall clearly indicate which methods are used for each component.

5.3.3 Class: Cryptographic Support (FCS)

FCS_VAL_EXT.1 Validation

FCS_VAL_EXT.1.1 The TSF shall perform validation of the [selection: submask, intermediate key, BEV] using the following method(s): [selection:

- key wrap as specified in FCS_COP.1(d);
- hash the [selection: submask, intermediate key, BEV] as specified in [selection: FCS_COP.1(b), FCS_COP.1(c)] and compare it to a stored hashed [selection: submask, intermediate key, BEV];
- decrypt a known value using the [selection: submask, intermediate key, BEV] as specified in FCS_COP.1(f) and compare it against a stored known value].

Application Note: *The EM Module performs validation of any administrator credential used to log in to the EM in accordance with this SFR.*

5.4 SFRs Defined for PP-Module

The following SFRs are required for an ST and TOE to conform to this cPP-Module. Conditional and strictly optional capabilities are defined in Appendix A and Appendix B.

5.4.1 Class: Cryptographic Support (FCS)

FCS_KYC_EXT.1/Server Key Chaining (Initiator) (Management Server)

FCS_KYC_EXT.1.1/Server The TSF shall maintain a key chain of: [selection:

- one, using a submask as the BEV;
- intermediate keys originating from one or more [selection: submask(s), recovery value(s)] to the [selection: BEV, enterprise server and from the enterprise server to the BEV] using the following method(s): [selection:
    - key derivation as specified in FCS_KDF_EXT.1,
    - key wrapping as specified in FCS_COP.1(d),
    - key combining as specified in FCS_SMC_EXT.1,
    - key transport as specified in FCS_COP.1(e),
    - key encryption as specified in FCS_COP.1(g)],

  and generated by the TSF using the following method(s): [selection:

    - asymmetric key generation as specified in FCS_CKM.1(a),
    - symmetric key generation as specified in FCS_CKM.1(b)];

while maintaining an effective strength of [selection: 128 bits, 256 bits] for symmetric keys and an effective strength of [selection: not applicable, 112 bits, 128 bits, 192 bits, 256 bits] for asymmetric keys.

Application Note: *The selections for the method of creating and maintaining the key chain are dependent on the second selection for intermediate keys. If the BEV is chosen as the selection, the key chain may be created and maintained by the AA. The ST Author should clearly indicate which portions of the key chain are created and maintained by the enterprise server and which are created and maintained by the AA.*

FCS_KYC_EXT.1.2/Server Refinement: The TSF shall provide a [selection: 128 bit, 256 bit] BEV to [*the EE*] [selection: after the TSF has successfully performed the validation process as specified in FCS_VAL_EXT.1, **after [*assignment: entity in the Operational Environment responsible for user authentication*] has successfully performed the user validation process as specified in FCS_VAL_EXT.2**].

Application Note: *This SFR is identical to its counterpart in the Base-PP except for the added ability to rely on the Operational Environment for validation (FCS_VAL_EXT.2) and the removal of this 'without validation taking place' selection option. Regardless of whether or not an endpoint maintains its own key chain, the Management Server must provide this* *functionality in order to protect the BEVs that it maintains. Note that the FCS_VAL_EXT.1 reference applies to the capability implemented in the Base-PP; the EM is not expected to enforce this.*

The Operational Environment in this instance refers to the native user authentication process used by the underlying OS of the user's system in a Remote Managed Environment such as Active Directory.

If validation by the Operational Environment is selected and local validation is also supported by the TOE when the Operational Environment resource is not available, both selections shall be made and the TSF shall comply with all requirements for both local validation as defined in FCS_VAL_EXT.1 in the Base-PP as well as those requirements for Operational Environment Validation as defined in FCS_VAL_EXT.2.

FCS_SMC_EXT.1/Server Submask Combining (Management Server)

FCS_SMC_EXT.1.1/Server The TSF shall combine submasks using the following method [selection: exclusive OR (XOR), SHA-256, SHA-384, SHA-512] to generate an [*intermediary key or BEV*].

Application Note: *This requirement specifies the way that a product may combine the various submasks by using either an XOR or an approved SHA-hash. The approved hash functions are captured in FCS_COP.1(b).*

5.4.2 Class: Identification and Authentication (FIA)

FIA_UAU.1 Timing of Authentication

FIA_UAU.1.1 Refinement: The TSF shall allow [*assignment: list of TSF-mediated actions*] on behalf of the **administrator** ~~user~~ to be performed before the **administrator** ~~user~~ is authenticated.

FIA_UAU.1.2 Refinement: The TSF shall require each **administrator** ~~user~~ to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that **administrator** ~~user~~.

FIA_UID.1 Timing of Identification

FIA_UID.1.1 Refinement: The TSF shall allow [*assignment: list of TSF-mediated actions*] on behalf of the **administrator** ~~user~~ to be performed before the **administrator** ~~user~~ is identified.

FIA_UID.1.2 Refinement: The TSF shall require each **administrator** ~~user~~ to be successfully identified before allowing any other TSF-mediated actions on behalf of that **administrator** ~~user~~.

5.4.3 Class: Security Management (FMT)

FMT_MTD.1 Management of TSF Data

FMT_MTD.1.1 Refinement: The TSF shall restrict the ability to [selection: change default, query, modify, delete, clear, [*assignment: other operations*]] the [*encryption keys and intermediate values*] to [*administrators*] **at the following times: [selection: never, during initial provisioning, during recovery]**.

FMT_SMF.1/Server Specification of Management Functions (Management Server)

FMT_SMF.1.1/Server Refinement: The TSF shall be capable of performing the following management functions: [***selection:***

- ***register new endpoint,***
- ***revoke registration of endpoint,***
- ***initiate key generation,***
- ***initiate key escrow,***
- ***initiate key zeroization,***
- ***initiate key recovery,***
- ***set encryption policy (supported algorithms and key sizes),***
- ***change administrator passwords***
- ***change user passwords,***
- ***change recovery credentials,***
- ***define administrators of the TOE,***
- ***enable/disable use of recovery credential,***
- ***configure number of failed authentication attempts before issuing a key sanitization of the DEK,***
- ***configure the number of authentication attempts that can be made within a 24 hour period,***
- ***configure the number of failed authentication attempts required to begin blocking subsequent attempts,***
- ***[assignment: ability to enable or disable one or more functions defined in the Base-PP],***
- ***[assignment: ability to perform one or more functions defined in the Base-PP],***
- ***[assignment: ability to authorize whether or not users can perform one or more functions defined in the Base-PP]***

].

Application Note: *This SFR refers specifically to the management functions that can be performed by the Management Server. Functions that are performed by the rest of the TOE are addressed by the FMT_SMF.1 SFR in the Base-PP. The final two assignments provide the ST author the ability to indicate when Base-PP functionality (such as configuration of power saving states) can be configured by the Management Server.*

The TSF's ability to initiate key generation, escrow, zeroization, and/or recovery may be accomplished either by the TOE performing those functions or by the TOE issuing a request to a remote endpoint to perform the functions. The ST author shall indicate which case is provided by the TSF. If the TOE performs any of the cryptographic functions that are selected as being *initiated in this SFR, the ST author shall include the equivalent FCS SFRs from the Base-PP as part of the TOE, specifically indicating that these functions are provided by the Management Server component of the TOE.*

If the TSF supports the use of a recovery credential (see Appendix B), the ST author shall include the 'enable/disable use of recovery credential' selection.

FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles [*administrator, user*].

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions [

- *the administrator role shall be able to administer the Management Server locally,*
- *the administrator role shall be able to administer the Management Server remotely,*
- *the administrator role shall be able to administer the endpoint(s) locally,*
- *the administrator role shall be able to administer the endpoint(s) remotely*

] are satisfied.

Application Note: *The intent of this SFR is to define a mechanism to distinguish administrators (who have the ability to configure the TSF and its data) from users (individuals in the enterprise who have FDEs on their systems). The TSF does not need to provide roles that are explicitly called 'administrator' or 'user'; the ST shall logically define the administrator as a combination of one or more roles that are provided by the TOE. A user as defined by this cPP-Module may be either a user that is specifically assigned an unprivileged role by the TSF or it may be characterized by an individual that lacks an administrator account on the TOE.*

The TSF may optionally provide the ability to rely on an external authentication mechanism to identify users in the case of a user requesting distribution of a recovery credential (see Appendix A.4). In this situation, the TOE's reliance on the Operational Environment is functionally equivalent to the TSF maintaining the user role as defined by FMT_SMR.2.1.

5.4.4 Class: Protection of the TSF (FPT)

FPT_ITT.1 Basic Internal TSF Data Transfer Protection

FPT_ITT.1.1 Refinement: The TSF shall protect TSF data from [disclosure, modification] when it is transmitted between separate parts of the TOE **through the use of [selection, choose at least one of: IPsec, SSH, TLS, TLS/HTTPS]**.

Application Note: *This SFR is intended to define protected communications between the Management Server (described by this cPP-Module) and remote Authorization Acquisition endpoints (described by the Base-PP).*

The TSF may rely on the Operational Environment to provide the cryptographic functionality that is used to establish these trusted communications. If the TOE implements its own cryptographic capability to perform this function, the ST author must claim the applicable

SFRs for cryptographic primitives and certificate validation from Appendix A.1 as well as the supported cryptographic protocol(s) from Appendix B.3.

FPT_KYP_EXT.2 Storage of Protected Key and Key Material

FPT_KYP_EXT.2.1 The TSF shall only store protected key and key material [<u>selection: within the TSF, in a SQL database in the Operational Environment, [*assignment: other key storage location*]</u>].

FPT_KYP_EXT.3 Attribution of Protected Key and Key Material

FPT_KYP_EXT.3.1 Refinement: The TSF shall maintain an association between [*BEV*, ***[<u>selection: key chain, no other key and key material</u>]***] and [*remote endpoints*, ***[selection: user identity, system identity, recovery credential, no other subjects]***].

Application Note: *The intent of this SFR is that at minimum, a BEV is associated with the drive(s) for which it was explicitly created by the TSF. If the TOE has the ability to maintain a key chain for a BEV, this SFR is intended to require an association between the key chain and BEV through the user account name and/or system name that is authorized to use the BEV. Likewise, if the TOE supports the use of a recovery credential, this SFR is intended to require an association between a BEV or key chain and the recovery credential used to recover that data.*

FPT_KYP_EXT.3.2 The TSF shall provide the ability to register remote endpoints by [*assignment: exchange of mutually identifying information that allows for an association to be made*].

Application Note: *The ST author will complete the assignment with information on the method used by the Management Server portion of the TOE to establish the association with the AA portion of the TOE described in FPT_KYP_EXT.3.1.*

FPT_KYP_EXT.3.3 The TSF shall provide the ability to revoke the registration of remote endpoints by [*assignment: method of removing and/or exchanging information that prevents further communications between the TOE and the endpoint*].

FPT_KYP_EXT.3.4 The TSF shall transmit any secure or private cryptographic information that is transferred between the TOE and a remote endpoint in order to establish or disestablish an association using a communications channel with a security strength at least as great as the strength of the information being transmitted.

Application Note: *The channel used to transmit this data is defined in FPT_ITT.1.*

5.4.5 Class: Trusted Path/Channels (FTP)

FTP_TRP.1 Trusted Path

FTP_TRP.1.1 Refinement: The TSF shall **be capable of using [<u>selection: *IPsec, SSH, TLS, HTTPS*</u>] to** provide a communication path between itself and **authorized remote administrators** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [modification, disclosure].

FTP_TRP.1.2 Refinement: The TSF shall permit **remote administrators** to initiate communication via the trusted path.

FTP_TRP.1.3 Refinement: The TSF shall require the use of the trusted path for **initial administrator authentication and all remote administration actions.**

Application Note: *This SFR is intended to define protected communications between the Management Server (described by this cPP-Module) and remote Authorization Acquisition endpoints (described by the Base-PP).*

The TSF may rely on the Operational Environment to provide the cryptographic functionality that is used to establish the trusted path. If the TOE implements its own cryptographic capability to perform this function, the ST author must claim the applicable SFRs for cryptographic primitives and certificate validation from Appendix A.1 as well as the supported cryptographic protocol(s) from Appendix B.3.

6. Security Assurance Requirements

This cPP-Module does not prescribe any SARs above and beyond what are required by the Base-PP, except that these SARs will apply to the entire TOE and not just to the functionality described by the Base-PP. [SD] includes Assurance Activities that are prescribed for this cPP-Module in order to show that the functionality defined in this cPP-Module satisfies the SARs for the supported PP-Configurations.

Appendix A: Optional Requirements

As indicated in the introduction to this cPP, the baseline requirements (those that must be performed by the TOE) are contained in the body of this cPP. Additionally, there are two other sets of requirements specified in Appendices A and B.

The first set (in this Appendix) are requirements that can be included in the ST, but do not have to be in order for a TOE to claim conformance to this cPP. The second set (in Appendix B) are requirements based on selections in the body of the cPP: if certain selections are made, then additional requirements in that appendix would need to be included in the body of the ST (e.g., cryptographic protocols selected in a trusted channel requirement).

A.1 Internal Cryptographic Implementation (Server Communications)

As indicated in the body of this cPP-Module, the functionality described by the cPP-Module requires a remote interface to the part of the TOE that is addressed by the Base-PP as well as a trusted path from a remote administrator to the TSF. Similar to the Base-PP, the Enterprise Management component (Management Server) may either provide its own internal cryptographic and signature services functionality or it may rely on this functionality to be provided by its Operational Environment. If the Management Server provides its own cryptographic functionality and/or signature services to support trusted communications, the applicable SFRs listed in this section shall be included in a conformant ST.

Note that these SFRs are all derived from the Base-PP but are iterated to reference server communications specifically. If the TSF provides two distinct cryptographic modules (one for intra-TOE communications and one for manipulation of the key chain used to protect the BEV), the ST author shall reference Appendix A.3 for guidance on how to document this in the ST.

FCS_CKM.1(a)/Server Cryptographic Key Generation (Asymmetric Keys) (Server Communications)

FCS_CKM.1.1(a)/Server Refinement: The TSF shall generate **asymmetric** cryptographic keys **for establishment of trusted channels and paths** in accordance with a specified cryptographic key generation algorithm: [***<u>selection:</u>***

- ***<u>RSA schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;</u>***
- ***<u>ECC schemes using "NIST curves" [selection: P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;</u>***
- ***<u>FFC schemes using cryptographic key sizes of [selection: 2048-bit, 3072-bit, 4096-bit] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1</u>***

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

Application Note: *The ST author selects all key generation schemes used for key establishment and device authentication. When key generation is used for key establishment, the schemes in FCS_CKM.2.1/Server and selected cryptographic protocols must match the selection. When key generation is used for device authentication, the public key is expected to be associated with an X.509v3 certificate.*

If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.

FCS_CKM.2/Server Cryptographic Key Establishment (Server Communications)

FCS_CKM.2.1/Server Refinement: The TSF shall **perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [***selection:***

- ***RSA-based key establishment schemes that meets the following: NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography";***
- ***Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***
- ***Finite field-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***
- ***AES-based key establishment schemes that meets the following: NIST Special Publication 800-38F "Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping";***
- ***Key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526, Section 3;***

] ~~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 9 of NIST SP 800-56B; however, Section 9 relies on implementation of other sections in SP 800-56B. If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation. The Suite B algorithms listed above (RFC 6460) are the preferred algorithms for implementation. It is recognized that RFC 5246 mandates the ciphersuite TLS_RSA_WITH_AES_128_CBC_SHA, but this ciphersuite is not tested with this requirement.

The elliptic curves used for the key establishment scheme correlate with the curves specified in FCS_CKM.1.1/Server.

The domain parameters used for the finite field-based key establishment scheme are specified by the key generation according to FCS_CKM.1.1/Server.

The AES-based key wrapping methods used as key transport scheme for key establishment are specified in FCS_COP.1(d)/Server.

FCS_CKM.4(a)/Server Cryptographic Key Destruction (Server Communications)

FCS_CKM.4.1(a)/Server Refinement: The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [***selection:***

- ***For volatile memory, the destruction shall be executed by a [selection:***
    - ***single direct overwrite [selection: consisting of a pseudo-random pattern using the TSF's RBG, consisting of zeroes, ones, a new value of a key, [assignment: some value that does not contain any CSP]];***
    - ***removal of power to the memory;***
    - ***destruction of reference to the key directly followed by a request for garbage collection];***
- ***For non-volatile memory [that consists of the invocation of an interface provided by the underlying platform that [selection:***
    - ***logically addresses the storage location of the key and performs a [selection: single, [assignment: ST author defined multi-pass]] direct 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]];***
    - ***instructs the underlying platform to destroy the abstraction that represents the key]***

]

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

Application Note: *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 visible. For instance, in a given implementation the application may have access to the file system details and may be able to logically address specific memory locations. In another implementation, the application may simply have a handle to a resource and can only ask the platform to delete the resource. The level of detail to which the TOE has access will be reflected in the TSS section of the ST.*

Several selections allow assignment of a 'value that does not contain any CSP'. This means that the TOE uses some other specified data not drawn from an RBG meeting FCS_RBG_EXT.1/Server requirements, and not being any of the particular values listed as other selection options. The point of the phrase 'does not contain any CSP' is to ensure that the overwritten data is carefully selected, and not taken from a general 'pool' that might contain current or residual data that itself requires confidentiality protection.

Key destruction does not apply to the public component of asymmetric key pairs.

FCS_COP.1(a)/Server Cryptographic Operation (Signature Generation and Verification) (Server Communications)

FCS_COP.1.1(a)/Server Refinement: The TSF shall perform [*cryptographic signature services (generation and verification)*] in accordance with a specified cryptographic algorithm [***selection:***

- ***RSA Digital Signature Algorithm and cryptographic key size (modulus) of 2048 bits or greater,***
- ***Elliptic Curve Digital Signature Algorithm and cryptographic key size of 256 bits or greater***

]

~~and cryptographic key sizes [assignment: *cryptographic key sizes*]~~ that meet the following: [***selection:***

- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1-v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3, for RSA schemes,***
- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [selection: P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4, for ECDSA schemes***

].

Application Note: *The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1(a). For example, SHA-256 should be chosen for 2048-bit RSA or ECC with P-256, SHA-384 should be chosen for 3072-bit RSA, 4096-bit RSA, or ECC with P-384, and SHA-512 should be chosen for ECC with P-521. The selection of the standard is made based on the algorithms selected.*

FCS_COP.1(b)/Server Cryptographic Operation (Hash Algorithm) (Server Communications)

FCS_COP.1.1(b)/Server Refinement: The TSF shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [***selection: SHA-256, SHA-384, SHA-512***] ~~and cryptographic key sizes [assignment: *cryptographic key sizes*]~~ that meet the following: [*ISO/IEC 10118-3:2004*].

Application Note: *The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1(f) and FCS_COP.1(a) (for example, SHA 256 for 128-bit keys).*

FCS_COP.1(c)/Server Cryptographic Operation (Keyed Hash Algorithm) (Server Communications)

FCS_COP.1.1(c)/Server Refinement: The TSF shall perform [*keyed-hash message authentication*] in accordance with a specified cryptographic algorithm [***selection: HMAC-***

__SHA-256, HMAC-SHA-384, HMAC-SHA-512__] and cryptographic key sizes [*assignment: cryptographic key size **(in bits) used in HMAC***] **and message digest sizes [*__selection: 256, 384, 512__*] *bits*** that meet the following: [*ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*].

Application Note: *The key size [k] in the assignment 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.*

FCS_COP.1(d)/Server Cryptographic Operation (Key Wrapping) (Server Communications)

FCS_COP.1.1(d)/Server Refinement: The TSF shall perform [*key wrapping*] in accordance with a specified cryptographic algorithm [*AES in the following modes **[__selection: KW, KWP, GCM, CCM__]***] and the cryptographic key size [***__selection: 128 bits, 256 bits__***] that meet the following: [*AES as specified in ISO/IEC 18033-3, **[__selection: NIST SP 800-38F, ISO/IEC 19772, no other standards__]***].

FCS_COP.1(e)/Server Cryptographic Operation (Key Transport) (Server Communications)

FCS_COP.1.1(e)/Server Refinement: The TSF shall perform [*key transport*] in accordance with a specified cryptographic algorithm [*RSA in the following modes **[__selection: KTS-OAEP, KTS-KEM-KWS__]***] and the cryptographic key size [***__selection: 2048 bits, 3072 bits__***] that meet the following: [*NIST SP 800-56B, Revision 1*].

Application Note: *This requirement is used in the body of the ST if the ST author chooses to use key transport in the key chaining approach that is specified in FCS_KYC_EXT.1/Server.*

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

FCS_COP.1.1(f)/Server Refinement: The TSF shall perform [*encryption/decryption]* in accordance with a specified cryptographic algorithm [*AES **used in [__selection: CBC, GCM__] mode***] and cryptographic key sizes [***__selection: 128 bits, 192 bits, 256 bits__***] that meet the following: [*AES as specified in ISO 18033-3, **[__selection: CBC as specified in ISO 10116, GCM as specified in ISO 19772__]***].

Application Note: *For the first selection of FCS_COP.1.1(f)/Server, the ST author should choose the mode or modes in which AES operates. For the second selection, the ST author should choose the key sizes that are supported by this functionality. The modes and key sizes selected here correspond to the cipher suite selections made in the trusted channel requirements.*

FCS_COP.1(g)/Server Cryptographic Operation (Key Encryption) (Server Communications)

FCS_COP.1.1(g)/Server Refinement: The TSF shall perform [*key encryption and decryption*] in accordance with a specified cryptographic algorithm [*AES **used in [__selection: CBC, GCM__] mode***] and cryptographic key sizes [***__selection: 128 bits, 256 bits__***] that meet the following: [*AES as specified in ISO /IEC 18033-3, **[__selection: CBC as specified in ISO/IEC 10116, GCM as specified in ISO/IEC 19772__]***].

Application Note: *This requirement is used in the body of the ST if the ST author chooses to use AES encryption/decryption for protecting the keys as part of the key chaining approach that is specified in FCS_KYC_EXT.1/Server.*

FCS_RBG_EXT.1/Server Random Bit Generation (Server Communications)

FCS_RBG_EXT.1.1/Server 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/Server The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [selection: [*assignment: number of software-based sources*] software-based noise source(s), [*assignment: number of hardware-based sources*] hardware-based noise source(s)] with a minimum of [selection: 128 bits, 192 bits, 256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for Hash Functions”, of the keys and hashes that it will generate.

Application Note: *For the first selection in FCS_RBG_EXT.1.2/Server, the ST selects at least one of the types of noise sources. If the TOE contains multiple noise sources of the same type, the ST author fills the assignment with the appropriate number for each type of source (e.g., 2 software-based noise sources, 1 hardware-based noise source). The documentation and tests required in the assurance activity for this element necessarily cover each source indicated in the ST.*

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). The ST author will select the function used, and include the specific underlying cryptographic primitives used in the requirement. While any of the identified hash functions (SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed.

FCS_SNI_EXT.1/Server Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation) (Server Communications)

FCS_SNI_EXT.1.1/Server The TSF shall only use salts that are generated by a [selection: DRBG as specified in FCS_RBG_EXT.1/Server, DRBG provided by the host platform].

FCS_SNI_EXT.1.2/Server The TSF shall only use unique nonces, with a minimum size of [*64*] bits.

FCS_SNI_EXT.1.3/Server The TSF shall create IVs in the following manner:

- CBC: IVs shall be non-repeating,
- CCM: Nonce shall be non-repeating.
- XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer,
- GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key.

Application Note: *This requirement covers several important factors – the salt must be random, but the nonces only have to be unique. FCS_SNI_EXT.1.3/Server specifies how the IV should be handled for each encryption mode. CBC, XTS, and GCM are allowed for AES encryption of the data. AES-CCM is an allowed mode for Key Wrapping.*

FIA_X509_EXT.1/Server X.509 Certificate Validation (Server Communications)

FIA_X509_EXT.1.1/Server The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The TSF shall validate the revocation status of the certificate using [selection: the Online Certificate Status Protocol (OCSP) as specified in RFC 2560, a Certificate Revocation List (CRL) as specified in RFC 5759].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
    - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

Application Note: *FIA_X509_EXT.1.1/Server lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. The trusted channel/path protocols require that certificates are used; this use requires that the extendedKeyUsage rules are verified.*

The validation is expected to end in a trusted root CA certificate in a root store managed by the platform.

FIA_X509_EXT.1.2/Server The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

FIA_X509_EXT.2/Server X.509 Certificate Authentication (Server Communications)

FIA_X509_EXT.2.1/Server The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication and [no additional uses].

FIA_X509_EXT.2.2/Server When the TSF cannot determine the validity of a certificate, the TSF shall [selection: allow the administrator to choose whether to accept the certificate in these cases, accept the certificate, not accept the certificate].

Application Note: *The certificate may be accepted by the TSF if there is another way to verify its validity. For example, the certificate may be considered trusted if found in the "Trusted Publishers" store or the certificate thumbprint was made known to the client out-of-band in advance for comparison.*

FIA_X509_EXT.3/Server X.509 Certificate Requests (Server Communications)

FIA_X509_EXT.3.1/Server The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [selection: device-specific information, Common Name, Organization, Organizational Unit, Country].

Application Note: *The public key is the public key portion of the public-private key pair generated by the TOE as specified in FCS_CKM.1(a)/Server.*

FIA_X509_EXT.3.2/Server The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

A.2 Internal Cryptographic Implementation (Key Attribution)

As stated in FPT_KYP_EXT.3, the TSF is expected to provide a method to uniquely associate cryptographic data with the subjects to which it applies. This is accomplished through the use of key distribution, which may be provided by the TOE or by a validated cryptographic module in the Operational Environment. If the TOE provides this cryptographic functionality, the following SFR shall be included in a conformant ST:

FCS_CKM.2 Cryptographic Key Distribution

FCS_CKM.2.1 Refinement: The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method: **[*selection:***

- ***RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography";***
- ***Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***
- ***Finite field-based key establishment schemes that meet the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"***

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

A.3 Internal Cryptographic Implementation (Server Management of Key Chain)

An enterprise deployment of full drive encryption capabilities may be designed such that the central management server is responsible for performing cryptographic functionality related to the creation and maintenance of a key chain which is then passed down to the individual endpoints rather than having each endpoint perform its own cryptographic functions. The PP-Configurations that this PP-module supports may include cryptographic SFRs that are defined in the Base-PP [FDE – AA]. If the Enterprise Management capability of the TOE is responsible for implementing this functionality, these SFRs can be included without modification, but the ST author must clearly note where in the TOE the claimed SFRs are enforced.

A.4 Configurable Encryption Policy

The TSF does not necessarily need to provide the ability to configure the behavior of the cryptographic functionality with respect to the cryptographic algorithms and key sizes that are used by the TOE. It is possible that the TSF has a single mode of operation that complies with the PP-Configuration, in which case no management of this functionality is required. If the TOE does provide this functionality, a compliant ST shall include the following SFR:

FMT_MOF.1/Server Management of Functions Behavior (Management Server)

FMT_MOF.1.1/Server Refinement: The TSF shall restrict the ability to [selection: determine the behaviour of, disable, enable, modify the behaviour of] the functions [***selection: encryption algorithms used, key sizes used***] to [*administrators*].

Application Note: *This SFR has been named with an iteration convention because the Base-PP defines a separate FMT_MOF.1 SFR for power management.*

Appendix B: Selection-Based Requirements

As indicated in the introduction to this cPP-Module, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this cPP-Module. There are additional requirements based on selections in the body of the cPP-Module: if certain selections are made, then additional requirements below may need to be included.

Note that many of these selection-based SFRs could also be implemented by cryptographic services in the TOE's Operational Environment. If this is the case, it is not necessary to include the SFRs in question so long as the Operational Environment can be shown to provide equivalent functionality.

B.1 Recovery Credentials

It is not mandatory for the TSF to provide the ability to support the use of recovery credentials. However, there are several SFRs (such as FCS_AFA_EXT.1) where the ST author can make a selection related to the use of recovery credentials. If any of these selections are made, the ST author shall include the threats, assumptions, OSPs, and environmental security objectives in this section. The ST author shall also include the SFR FIA_REC_EXT.1 along with the appropriate SFR(s) for the specific recovery credential type(s) supported by the TSF. Additionally, the ST author shall indicate in FMT_SMF.1/Server that the use of recovery credentials can be enabled and disabled by an administrator.

(T.RECOVERY_KEY_CHAIN_EXHAUST) The cPP-Module addresses the threat of an attacker taking advantage of a weak remote recovery algorithm to brute force attack the recovery key chain.

[Mandatory SFRs: None;

Optional SFRs: FIA_CHR_EXT.1, FIA_PIN_EXT.1, FIA_REC_EXT.1]

Rationale: Each method of recovery credentials [FIA_REC_EXT.1] provides security against exhaustion. The challenge/response method [FIA_CHR_EXT.1] is limited to the user requiring access to a specific system that they are attempting to access which means there is no threat of disclosure of the response being used as an attack vector against other users and devices. PINs [FIA_PIN_EXT.1] protected against an exhaustive brute force attack by only allowing a value to work once. Further brute force guessing is virtually impossible if the Management Server implements a periodic random PIN generation function such that it is virtually impossible for an attacker to successfully guess the PIN in the time window before the PIN is updated.

(T.REPLAY_RECOVERY_INFORMATION) The cPP-Module addresses the threat of an attacker replaying recovery information to gain access to the BEV, either because the communications channel used to transmit recovery information is insecure or because the recovery credential is not implemented as one-time use.

[Mandatory SFRs: FPT_ITT.1;

Optional SFRs: FIA_CHR_EXT.1, FIA_PIN_EXT.1]

Rationale: FPT_CHR_EXT.1 defines a mechanism for generating a one-time use recovery credential. FPT_ITT.1 defines a secure channel that will not subject its data in transit to loss of confidentiality or integrity. Therefore, any transmission of recovery credential data between TOE components will not be subjected to unauthorized modification or disclosure. FIA_PIN_EXT.1 provides an additional layer of logical security by placing limitations on when the recovery credential is valid. These limitations mean that it is of no benefit for an attacker to find an old PIN credential because their use was limited to a single instance or to a specific session that has since expired.

(A.TRAINED_USER/SERVER) This assumption extends the A.TRAINED_USER assumption in the Base-PP to assume that users are capable of interpreting and using recovery tokens provided by Authorized Administrators.

[OE.TRAINED_USER/SERVER]

(A.VERIFIED_USER) Administrators are assumed to validate the legitimacy of recovery requests before transmitting any recovery credentials to end users.

[OE.VERIFIED_USER]

(P.STRONG_PASSWORDS) The organization shall require that any recovery credentials that are created by a user adhere to the same password strength policy as the actual user passwords.

[OE.STRONG_PASSWORDS]

(OE.TRAINED_USER/SERVER) Authorized users will be properly trained and follow all guidance for securing any recovery credentials that are provided to them.

Rationale: Proper handling of recovery credentials is necessary to ensure that they are not subject to unauthorized disclosure and used in a timely manner.

(OE.VERIFIED_USER) An administrator will not release a recovery credential to a user unless the administrator is able to verify the legitimacy of the request.

Rationale: Technical controls that prevent unauthorized disclosure of a recovery credential can be negated through a social engineering attack. The TSF cannot provide a countermeasure to this so it is expected to be mitigated by the Operational Environment.

(OE.STRONG_PASSWORDS) User passwords and recovery credentials will adhere to the same level of password complexity such that an easily-guessed recovery credential does not allow for bypass of the password mechanism.

Rationale: The recovery credential and user password may be defined using different products and stored in different repositories, each of which may potentially define their own strength of secrets policies. If the strength of the recovery credential is weaker than the strength of the user password, an attacker can potentially leverage the weaker credential to cause a compromise of user data without having to attack a strong password mechanism.

(A.RECOVERY_CREDENTIAL_STRENGTH) Recovery credentials created by an end user are assumed to be at least as strong as the standard password, if used.

[OE.RECOVERY_CREDENTIAL_STRENGTH]

FIA_CHR_EXT.1 Challenge/Response Recovery Credential

FIA_CHR_EXT.1.1 The TSF shall only generate a response if it is able to access recovery information for [selection: the user requesting the recovery, the user requesting recovery and the device for which the recovery was requested].

Application Note: *This requires that the TSF has the ability to attribute the BEV and/or key chain information to the appropriate endpoint.*

FIA_CHR_EXT.1.2 The response shall only work on the system upon which the challenge was generated and the user to whom it was generated.

Application Note: *This mechanism is intended to provide a recovery method for a user who has forgotten their authentication factor and is unable to access their encrypted data on a system that is fully functional.*

FIA_CHR_EXT.1.3 The response shall only be used during the same session in which the request was generated.

Application Note: *The intent of this requirement is to limit the attack surface of the recovery credential mechanism by preventing the use of the credential following a reboot of the device.*

FIA_CHR_EXT.1.4 The TSF shall generate an ephemeral response that has at least as many potential values as a corresponding password or PIN.

FIA_CHR_EXT.1.5 The TSF shall allow a maximum of [*assignment: integer value*] of response entry attempts per boot cycle.

FIA_CHR_EXT.1.6 The TSF shall [selection:

- perform a key sanitization of the DEK upon [*assignment: ST author specified number or configurable range of attempts*] consecutive failed validation attempts,
- institute a delay such that only [*assignment: ST author specified number or configurable range of attempts*] validation attempts can be made within a 24 hour period,
- block validation after [*assignment: ST author specified number or configurable range of attempts*] of consecutive failed validation attempts,
- terminate the session after [*assignment: ST author specified number or configurable range of attempts*] consecutive failed validation attempts].

FIA_PIN_EXT.1 PIN Recovery Credential

FIA_PIN_EXT.1.1 The TSF shall pre-populate the recovery PIN on the Management Server.

FIA_PIN_EXT.1.2 The recovery key chain accessed by the recovery PIN shall only work on the system within which the drive or set of drives to be recovered resides.

FIA_PIN_EXT.1.3 The TSF shall not permit the PIN to be used more than once.

FIA_REC_EXT.1 Support for Recovery Credentials

FIA_REC_EXT.1.1 The TSF shall support the following recovery credentials: [selection: challenge/response, PIN].

FIA_REC_EXT.1.2 The TSF shall provide the ability to enable and disable the use of recovery credentials.

B.2 User Validation

The ST author must include this selection in the ST when the selection item pertaining to the Operational Environment is chosen as the validation method in FCS_KYC_EXT.1.2/Server.

FCS_VAL_EXT.2 User Validation

FCS_VAL_EXT.2.1 The TSF shall perform validation of the [user] by receiving assertion of the user's validity from: [*assignment: Operational Environment component responsible for user authentication*].

Application Note: *The ST author will specify a logical component in the Operational Environment that is capable of asserting a user's identity to the TOE, such as Active Directory.*

FCS_VAL_EXT.2.2 The TSF shall require validation of the user prior to [*transmitting BEV to the endpoint*].

FCS_VAL_EXT.2.3 The TSF shall [selection: [*assignment: key sanitization activity*] upon receiving a configurable number of consecutive failed validation attempts from the Operational Environment; institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period; block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts; require power cycle of or reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

Application Note: *If the local key chain exists when a BEV is present on the Management Server, the local key chain shall satisfy the key chaining requirements (including any related dependencies) as defined in the Base-PP.*

B.3 Cryptographic Protocols

This cPP-Module introduces the requirement for the TSF to provide trusted communications channels between distributed parts of the TOE (FPT_ITT.1) and from a remote administrator to the TOE (FTP_TRP.1). However, the specific cryptographic protocol(s) used to accomplish these is not mandated; any of IPsec, SSH, TLS, or TLS/HTTPS can be used. Based on the cryptographic protocol(s) implemented by the TSF to secure these communications, the ST author shall include at least one of the SFRs defined in this section. This section also includes SFRs that are optionally used in support of key chaining.

FCS_CKM.1(b)/Server Cryptographic Key Generation (Symmetric Keys)

FCS_CKM.1.1(b)/Server Refinement: The TSF shall generate **symmetric** cryptographic keys **using a Random Bit Generator as specified in FCS_RBG_EXT.1/Server** and specified cryptographic key sizes [**selection: 128 bit, 256 bit**] that meet the following: [*no standard*].

Application Note: *Symmetric keys may be used to generate keys along the key chain. Therefore, the ST author should select FCS_CKM.1(b)/Server, if Symmetric key generation is used.*

FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

Application Note: *The ST author must provide enough detail to determine how the implementation is complying with the standard(s) identified; this can be done either by adding elements to this component, or by additional detail in the TSS.*

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 The TSF shall [selection: not establish the connection, request authorization to establish the connection, no other action] if the peer certificate is deemed invalid.

Application Note: *Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.*

FCS_IPSEC_EXT.1 IPsec Protocol

FCS_IPSEC_EXT.1.1 The TSF shall implement the IPsec architecture as specified in RFC 4301.

Application Note: *RFC 4301 calls for an IPsec implementation to protect IP traffic through the use of a Security Policy Database (SPD). The SPD is used to define how IP packets are to be handled: PROTECT the packet (e.g., encrypt the packet), BYPASS the IPsec services (e.g., no encryption), or DISCARD the packet (e.g., drop the packet). The SPD can be implemented in various ways, including router access control lists, firewall rulesets, a "traditional" SPD, etc. Regardless of the implementation details, there is a notion of a "rule" that a packet is "matched" against and a resulting action that takes place.*

While there must be a means to order the rules, a general approach to ordering is not mandated, as long as the SPD can distinguish the IP packets and apply the rules accordingly. There may be multiple SPDs (one for each network interface), but this is not required.

FCS_IPSEC_EXT.1.2 The TSF shall have a nominal, final entry in the SPD that matches anything that is otherwise unmatched, and discards it.

FCS_IPSEC_EXT.1.3 The TSF shall implement transport mode and [selection: tunnel mode, no other mode].

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms AES-CBC-128, AES-CBC-256 (both specified by RFC 3602) and [selection: AES-GCM-128 (specified in RFC 4106), AES-GCM-256 (specified in RFC 4106), no other algorithms] together with a Secure Hash Algorithm (SHA)-based HMAC.

FCS_IPSEC_EXT.1.5 The TSF shall implement the protocol: [selection:

- IKEv1, using Main Mode for Phase 1 exchanges, as defined in RFCs 2407, 2408, 2409, RFC 4109, [selection: no other RFCs for extended sequence numbers, RFC 4304 for extended sequence numbers], and [selection: no other RFCs for hash functions, RFC 4868 for hash functions];
- IKEv2 as defined in RFC 5996 and [selection: with no support for NAT traversal, with mandatory support for NAT traversal as specified in RFC 5996, section 2.23)], and [selection: no other RFCs for hash functions, RFC 4868 for hash functions]

].

Application Note: *If the TOE implements SHA-2 hash algorithms for IKEv1 or IKEv2, the ST author selects RFC 4868. If the ST author selects IKEv1, FCS_IPSEC_EXT.1.15 must also be included in the ST. IKEv2 will be required for those TOEs entering evaluation after Quarter 3, 2016.*

FCS_IPSEC_EXT.1.6 The TSF shall ensure the encrypted payload in the [selection: IKEv1, IKEv2] protocol uses the cryptographic algorithms AES-CBC-128, AES-CBC-256 as specified in RFC 3602 and [selection: AES-GCM-128, AES-GCM-256 as specified in RFC 5282, no other algorithm].

Application Note: *AES-GCM-128 and AES-GCM-256 may only be selected if IKEv2 is also selected, as there is no RFC defining AES-GCM for IKEv1.*

FCS_IPSEC_EXT.1.7 The TSF shall ensure that [selection:

- IKEv1 Phase 1 SA lifetimes can be configured by an administrator based on [selection:
  - number of bytes;
  - length of time, where the time values can configured within [*assignment: integer range including 24*] hours;

  ];

- IKEv2 SA lifetimes can be configured by an administrator based on [selection:
  - number of bytes;
  - length of time, where the time values can configured within [*assignment: integer range including 24*] hours

]

].

Application Note: *The ST author chooses either the IKEv1 requirements or IKEv2 requirements (or both, depending on the selection in FCS_IPSEC_EXT.1.5). The ST author chooses either volume-based lifetimes or time-based lifetimes (or a combination). This requirement must be accomplished by providing Security Administrator-configurable lifetimes (with appropriate instructions in documents mandated by AGD_OPE). Hardcoded limits do not meet this requirement. In general, instructions for setting the parameters of the implementation, including lifetime of the SAs, should be included in the guidance documentation generated for AGD_OPE.*

FCS_IPSEC_EXT.1.8 The TSF shall ensure that [selection:

- IKEv1 Phase 2 SA lifetimes can be configured by an administrator based on [selection:
  - number of bytes;
  - length of time, where the time values can be configured within [*assignment: integer range including 8*] hours;

  ];
- IKEv2 Child SA lifetimes can be configured by an administrator based on [selection:
  - number of bytes;
  - length of time, where the time values can be configured within [*assignment: integer range including 8*] hours;

  ]

].

Application Note: *The ST author chooses either the IKEv1 requirements or IKEv2 requirements (or both, depending on the selection in FCS_IPSEC_EXT.1.5). The ST author chooses either volume-based lifetimes or time-based lifetimes (or a combination). This requirement must be accomplished by providing Security Administrator-configurable lifetimes (with appropriate instructions in documents mandated by AGD_OPE). Hardcoded limits do not meet this requirement. In general, instructions for setting the parameters of the implementation, including lifetime of the SAs, should be included in the guidance documentation generated for AGD_OPE.*

FCS_IPSEC_EXT.1.9 The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange (“x” in g^x mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least [*assignment: (one or more) number(s) of bits that is at least twice the security strength of the negotiated Diffie-Hellman group*] bits.

Application Note: *For DH groups 19 and 20, the "x" value is the point multiplier for the generator point G.*

Since the implementation may allow different Diffie-Hellman groups to be negotiated for use in forming the SAs, the assignment in FCS_IPSEC_EXT.1.9 may contain multiple values. For each DH group supported, the ST author consults Table 2 in NIST SP 800-57

"Recommendation for Key Management –Part 1: General" to determine the security strength ("bits of security") associated with the DH group. Each unique value is then used to fill in the assignment for this element. For example, suppose the implementation supports DH group 14 (2048-bit MODP) and group 20 (ECDH using NIST curve P-384). From Table 2, the bits of security value for group 14 is 112, and for group 20 it is 192.

FCS_IPSEC_EXT.1.10 The TSF shall generate nonces used in [selection: IKEv1, IKEv2] exchanges of length [selection:

- [*assignment: security strength associated with the negotiated Diffie-Hellman group*];
- at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash

].

Application Note: *The ST author must select the second option for nonce lengths if IKEv2 is also selected (as this is mandated in RFC 5996). The ST author may select either option for IKEv1.*

For the first option for nonce lengths, since the implementation may allow different Diffie-Hellman groups to be negotiated for use in forming the SAs, the assignment in FCS_IPSEC_EXT.1.10 may contain multiple values. For each DH group supported, the ST author consults Table 2 in NIST SP 800-57 "Recommendation for Key Management –Part 1: General" to determine the security strength ("bits of security") associated with the DH group. Each unique value is then used to fill in the assignment for this element. For example, suppose the implementation supports DH group 14 (2048-bit MODP) and group 20 (ECDH using NIST curve P-384). From Table 2, the bits of security value for group 14 is 112, and for group 20 it is 192.

Because nonces may be exchanged before the DH group is negotiated, the nonce used should be large enough to support all TOE-chosen proposals in the exchange.

FCS_IPSEC_EXT.1.11 The TSF shall ensure that all IKE protocols implement DH Groups 14 (2048-bit MODP), and [selection: 19 (256-bit Random ECP), 5 (1536-bit MODP), 24 (2048-bit MODP with 256-bit POS), 20 (384-bit Random ECP), no other DH groups].

Application Note: *The selection is used to specify additional DH groups supported. This applies to IKEv1 and IKEv2 exchanges. For products entering into evaluation after Quarter 3, 2015, DH Group 19 (256-bit Random ECP) and DH Group 20 (384-bit Random ECP) will be required. It should be noted that if any additional DH groups are specified, they must comply with the requirements (in terms of the ephemeral keys that are established) listed in FCS_CKM.1.*

FCS_IPSEC_EXT.1.12 The TSF shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [selection: IKEv1 Phase 1, IKEv2 IKE_SA] connection is greater than or equal to the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [selection: IKEv1 Phase 2, IKEv2 CHILD_SA] connection.

Application Note: *The ST author chooses either or both of the IKE selections based on what is implemented by the TOE. Obviously, the IKE version(s) chosen should be consistent not only in this element, but with other choices for other elements in this component. While it is acceptable for this capability to be configurable, the default configuration in the evaluated configuration (either "out of the box" or by configuration guidance in the AGD documentation) must enable this functionality.*

FCS_IPSEC_EXT.1.13 The TSF shall ensure that all IKE protocols perform peer authentication using [selection: RSA, ECDSA] that use X.509v3 certificates that conform to RFC 4945 and [selection: pre-shared keys, no other method].

Application Note: *At least one public-key-based Peer Authentication method is required in order to conform to this PP; one or more of the public key schemes is chosen by the ST author to reflect what is implemented. The ST author also ensures that appropriate FCS requirements reflecting the algorithms used (and key generation capabilities, if provided) are listed to support those methods. Note that the TSS will elaborate on the way in which these algorithms are to be used (for example, RFC 2409 specifies three authentication methods using public keys; each one supported will be described in the TSS).*

FCS_IPSEC_EXT.1.14 The TSF shall only establish a trusted channel to peers with valid certificates.

Application Note: *Supported peer certificate algorithms are the same as FCS_IPSEC_EXT.1.1.*

FCS_KDF_EXT.1/Server Cryptographic Key Derivation (Management Server)

FCS_KDF_EXT.1.1/Server The TSF shall accept [selection: a RNG generated submask as specified in FCS_RBG_EXT.1/Server, a conditioned password submask, imported submask] to derive an intermediate key, as defined in [selection:

- NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF in Double-Pipeline Iteration Mode],
- NIST SP 800-132],

using the keyed-hash functions specified in FCS_COP.1(c)/Server, such that the output is at least of equivalent security strength (in number of bits) to the BEV.

Application Note: *This requirement is used in the body of the ST if the ST author chooses to use key derivation in the key chaining approach that is specified in FCS_KYC_EXT.1.*

This requirement establishes acceptable methods for generating a new random key or an existing submask to create a new key along the key chain.

FCS_PCC_EXT.1/Server Cryptographic Password Construct and Conditioning (Management Server)

FCS_PCC_EXT.1.1/Server A password used by the TSF to generate a password authorization factor shall enable up to [*assignment: positive integer of 64 or more*] characters in the set of {upper case characters, lower case characters, numbers, and [*assignment: other supported special* *characters*]} and shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm HMAC-[selection: SHA-256, SHA-512], with [*assignment: positive integer of 1000 or more*] iterations, and output cryptographic key sizes [selection: 128 bits, 256 bits] that meet the following: [*NIST SP 800-132*].

Application Note: *The admin password is represented on the administrator's machine as a sequence of characters whose encoding depends on the TOE and the underlying OS. This sequence must be conditioned into a string of bits that forms the submask to be used as input into the key chain. Conditioning can be performed using one of the identified hash functions or the process described in NIST SP 800-132; the method used is selected by the ST author. If 800-132 conditioning is specified, then the ST author fills in the number of iterations that are performed. 800-132 also requires the use of a pseudo-random function (PRF) consisting of HMAC with an approved hash function. The ST author selects the hash function used which also includes the appropriate requirements for HMAC.*

FCS_SSHC_EXT.1 SSH Client Protocol

FCS_SSHC_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and [selection: 5647, 5656, 6187, 6668, no other RFCs].

Application Note: *The ST author selects which of the additional RFCs to which conformance is being claimed. Note that these need to be consistent with selections in later elements of this component (e.g., cryptographic algorithms permitted). RFC 4253 indicates that certain cryptographic algorithms are "REQUIRED". This means that the implementation must include support, not that the algorithms must be enabled for use. Ensuring that algorithms indicated as "REQUIRED" but not listed in the later elements of this component are implemented is out of scope of the assurance activity for this requirement.*

FCS_SSHC_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based.

FCS_SSHC_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*assignment: number of bytes*] bytes in an SSH transport connection are dropped.

Application Note: *RFC 4253 provides for the acceptance of "large packets" with the caveat that the packets should be of "reasonable length" or dropped. The assignment should be filled in by the ST author with the maximum packet size accepted, thus defining "reasonable length" for the TOE.*

FCS_SSHC_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-cbc, aes256-cbc, [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other algorithms].

Application Note: *RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as encryption algorithms when the same algorithm is being used as the MAC algorithm. In the assignment, the ST author can select the AES-GCM algorithms, or "no other algorithms" if AES-GCM is not supported. If AES-GCM is selected, there should be corresponding FCS_COP entries in the ST.*

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH transport implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

Application Note: *Implementations that select only ssh-rsa will not achieve the 112-bit security strength in the digital signature generation for SSH authentication as is recommended in NIST SP 800-131A. Future versions of this profile may remove ssh-rsa as a selection. If x509v3-ecdsa-sha2-nistp256 or x509v3-ecdsa-sha2-nistp384 are selected, then the list of trusted certification authorities must be selected in FCS_SSHC_EXT.1.9.*

FCS_SSHC_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [selection: hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512] and [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other MAC algorithms] as its data integrity MAC algorithm(s) and rejects all other MAC algorithm(s).

Application Note: *RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as MAC algorithms when the same algorithm is being used as the encryption algorithm. RFC 6668 specifies the use of the sha2 algorithms in SSH.*

FCS_SSHC_EXT.1.7 The TSF shall ensure that [selection: diffie-hellman-group14-sha1, ecdh-sha2-nistp256] and [selection: ecdh-sha2-nistp384, ecdh-sha2-nistp521, no other methods] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHC_EXT.1.8 The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

FCS_SSHC_EXT.1.9 The TSF shall ensure that the SSH client authenticates the identity of the SSH server using a local database associating each host name with its corresponding public key or [selection: a list of trusted certification authorities, no other methods] as described in RFC 4251 section 4.1.

Application Note: *The list of trusted certification authorities can only be selected if x509v3-ecdsa-sha2-nistp256 or x509v3-ecdsa-sha2-nistp384 are selected in FCS_SSHC_EXT.1.5.*

FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and [selection: 5647, 5656, 6187, 6668, no other RFCs].

Application Note: *The ST author selects which of the additional RFCs to which conformance is being claimed. Note that these need to be consistent with selections in later elements of this component (e.g., cryptographic algorithms permitted). RFC 4253 indicates that certain cryptographic algorithms are "REQUIRED". This means that the implementation must include support, not that the algorithms must be enabled for use. Ensuring that algorithms indicated as "REQUIRED" but not listed in the later elements of this component are implemented is out of scope of the assurance activity for this requirement.*

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based.

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*assignment: number of bytes*] bytes in an SSH transport connection are dropped.

Application Note: *RFC 4253 provides for the acceptance of "large packets" with the caveat that the packets should be of "reasonable length" or dropped. The assignment should be filled in by the ST author with the maximum packet size accepted, thus defining "reasonable length" for the TOE.*

FCS_SSHS_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: *aes128-cbc, aes256-cbc,* [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other algorithms].

Application Note: *RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as encryption algorithms when the same algorithm is being used as the MAC algorithm. In the assignment, the ST author can select the AES-GCM algorithms, or "no other algorithms" if AES-GCM is not supported. If AES-GCM is selected, there should be corresponding FCS_COP entries in the ST.*

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH transport implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

Application Note: *Implementations that select only ssh-rsa will not achieve the 112-bit security strength in the digital signature generation for SSH authentication as is recommended in NIST SP 800-131A. Future versions of this profile may remove ssh-rsa as a selection.*

FCS_SSHS_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [selection: hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512] and [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other MAC algorithms] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

Application Note: *RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as MAC algorithms when the same algorithm is being used as the encryption algorithm. RFC 6668 specifies the use of the sha2 algorithms in SSH.*

FCS_SSHS_EXT.1.7 The TSF shall ensure that [selection: diffie-hellman-group14-sha1, ecdh-sha2-nistp256] and [selection: ecdh-sha2-nistp384, ecdh-sha2-nistp521, no other methods] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8 The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

FCS_TLSC_EXT.1 TLS Client Protocol

FCS_TLSC_EXT.1.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

Application Note: *The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the optional ciphersuites that are supported. It is necessary to limit the ciphersuites that can be used in an evaluated configuration administratively on the server in the test environment. The Suite B algorithms listed above (RFC 6460) are the preferred algorithms for implementation. It is recognized that RFC 5246 mandates the ciphersuite TLS_RSA_WITH_AES_128_CBC_SHA, but this ciphersuite is not tested with this requirement.*

In a future version of this cPP TLS v1.2 will be required for all TOEs.

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125.

Application Note: *The rules for verification of identify are described in Section 6 of RFC 6125. The reference identifier is established by the user (e.g. entering a URL into a web browser or clicking a link), by configuration (e.g. configuring the name of a mail server or authentication server), or by an application (e.g. a parameter of an API) depending on the application service. Based on a singular reference identifier's source domain and application service type (e.g. HTTP, SIP, LDAP), the client establishes all reference identifiers which are acceptable, such as a Common Name for the Subject Name field of the certificate and a (case-insensitive) DNS* *name, URI name, and Service Name for the Subject Alternative Name field. The client then compares this list of all acceptable reference identifiers to the presented identifiers in the TLS server's certificate.*

The preferred method for verification is the Subject Alternative Name using DNS names, URI names, or Service Names. Verification using the Common Name is required for the purposes of backwards compatibility. Additionally, support for use of IP addresses in the Subject Name or Subject Alternative name is discouraged as against best practices but may be implemented. Finally, the client should avoid constructing reference identifiers using wildcards. However, if the presented identifiers include wildcards, the client must follow the best practices regarding matching; these best practices are captured in the assurance activity.

FCS_TLSC_EXT.1.3 The TSF shall only establish a trusted channel if the peer certificate is valid.

Application Note: *Validity is determined by the identifier verification, certificate path, the expiration date, and the revocation status in accordance with RFC 5280. Certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/Server.*

FCS_TLSC_EXT.1.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1, or none] and no other curves.

Application Note: *If ciphersuites with elliptic curves were selected in FCS_TLSC_EXT.1.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_TLSC_EXT.1.1, then 'none' should be selected.*

This requirement limits the elliptic curves allowed for authentication and key agreement to the NIST curves from FCS_COP.1(a)/Server, FCS_CKM.1(a)/Server, and FCS_CKM.2/Server. This extension is required for clients supporting Elliptic Curve ciphersuites.

FCS_TLSC_EXT.3 TLS Client Handshake Message Exchange

FCS_TLSC_EXT.3.1 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

Application Note*: This selection is dependent on choosing TLS protocol in FPT_ITT.1 and TLS-PSK ciphersuite in FCS_TLSC_EXT.1.1. When a TSF uses these selections, a single symmetric encryption based ciphersuite means that there is no need to either negotiate the cryptographic algorithms or provide additional information to set the premaster secret or change the ciphersuite. In which case the "ClientHello" message is extended to include the PSK identity and the "Finished" message is sent immediately after the "ServerHello" message is received. No other messages are sent.*

FCS_TLSS_EXT.1 TLS Server Protocol

FCS_TLSS_EXT.1.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

Application Note: *The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the optional ciphersuites that are supported. It is necessary to limit the ciphersuites that can be used in an evaluated configuration administratively on the server in the test environment. The Suite B algorithms listed above (RFC 6460) are the preferred algorithms for implementation. It is recognized that RFC 5246 mandates the ciphersuite TLS_RSA_WITH_AES_128_CBC_SHA, but this ciphersuite is not tested with this requirement.*

In a future version of this cPP TLS v1.2 will be required for all TOEs.

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting SSL 1.0, SSL 2.0, SSL 3.0, TLS 1.0, and [selection: TLS 1.1, TLS 1.2, none].

Application Note: *All SSL versions and TLS v1.0 are denied. Any TLS versions not selected in FCS_TLSS_EXT.1.1 should be selected here.*

FCS_TLSS_EXT.1.3 The TSF shall generate key establishment parameters using RSA with key size 2048 bits and [selection: 3072 bits, 4096 bits, no other size] and [selection: over NIST curves [selection: secp256r1, secp384r1] and no other curves; Diffie-Hellman parameters of size 2048 bits and [selection: 3072 bits, no other size]; no other].

Application Note: *If the ST lists a DHE or ECDHE ciphersuite in FCS_TLSS_EXT.1.1, the ST must include the Diffie-Hellman or NIST curves selection in the requirement. FMT_SMF.1 requires the configuration of the key agreement parameters in order to establish the security strength of the TLS connection.*

FCS_TLSS_EXT.3 TLS Server Handshake Message Exchange

FCS_TLSS_EXT.3.1 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

Application Note*: This selection is dependent on choosing TLS protocol in FPT_ITT.1 and TLS-PSK ciphersuite in FCS_TLSS_EXT.1.1. When the TSF uses these selections, a single symmetric encryption based ciphersuite means that there is no need to either negotiate the cryptographic algorithms or provide additional information to set the premaster secret or change the ciphersuite. In which case the PSK identity is received in the "ClientHello" message and the server sends the "Finished" message immediately after the "ServerHello" message is sent. No other messages are sent.*

Appendix C: Extended Component Definitions

This appendix contains the definitions for the extended requirements that are used in the cPP, including those used in Appendices A and B.

Note that several of the extended requirements used for this cPP have dependencies on SFRs that are iterated in the cPP (e.g. FCS_COP.1(d)/Server). The reader is advised that the SFR names for these dependencies may differ if the same extended components are used in other Protection Profiles.

C.1 Background and Scope

This document provides a definition for all of the extended components introduced in this cPP-Module. These components are identified in the following table:

Table 3: Extended Components

Functional Class	Functional Components
Cryptographic Support (FCS)	FCS_HTTPS_EXT HTTPS Protocol FCS_IPSEC_EXT IPsec Protocol FCS_KDF_EXT Cryptographic Key Derivation FCS_KYC_EXT Key Chaining FCS_PCC_EXT Cryptographic Password Construct and Conditioning FCS_RBG_EXT Random Bit Generation FCS_SMC_EXT Submask Combining FCS_SNI_EXT Salt, Nonce, and Initialization Vector Generation FCS_SSHC_EXT SSH Client Protocol FCS_SSHS_EXT SSH Server Protocol FCS_TLSC_EXT TLS Client Protocol FCS_TLSS_EXT TLS Server Protocol FCS_VAL_EXT Validation of Cryptographic Elements
Identification and Authentication (FIA)	FIA_CHR_EXT Challenge/Response Recovery Credential FIA_PIN_EXT PIN Recovery Credential FIA_REC_EXT Support for Recovery Credentials FIA_X509_EXT Authentication Using X.509 Certificates
Protection of the TSF (FPT)	FPT_KYP_EXT Key and Key Material Protection

Note that there are several SFRs included in this cPP-Module that are iterations of extended SFRs that are defined in the Base-PP; the extended components definitions are not repeated here since they are considered to be part of the PP-Configuration. Likewise, the Base-PP includes several extended SFRs that may also apply to the Management Server in the PP-Configuration; these SFRs are similarly considered to be defined in the PP-Configuration through their definitions in the Base-PP. The extended components definition in this cPP-Module is limited to requirements that apply specifically to the Management Server and not to the part of the TOE described by the Base-PP.

C.2 Extended Component Definitions

FCS_HTTPS_EXT HTTPS Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using HTTPS.

Component Leveling

FCS_HTTPS_EXT.1, HTTPS Protocol, requires the TSF to implement HTTPS in accordance with RFC 2818 in a manner that supports TLS.

Management: FCS_HTTPS_EXT.1

No specific management functions are identified.

Audit: FCS_HTTPS_EXT.1

There are no auditable events foreseen.

FCS_HTTPS_EXT.1 HTTPS Protocol

Hierarchical to: No other components

Dependencies: FCS_TLSS_EXT.1 TLS Server Protocol,
FIA_X509_EXT.1 X.509 Certificate Validation

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 The TSF shall [selection: not establish the connection, request authorization to establish the connection, no other action] if the peer certificate is deemed invalid.

FCS_IPSEC_EXT IPsec Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using IPsec.

Component Leveling

FCS_IPSEC_EXT.1, IPsec Protocol, requires the TSF to implement IPsec in accordance with a specific manner.

Management: FCS_IPSEC_EXT.1

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

- Maintenance of SA lifetime configuration
- Specification of supported modes, cryptographic algorithms, and DH groups
- Configuration of IKE peer authentication method

Audit: FCS_IPSEC_EXT.1

The following actions should be auditable if FAU_GEN Security Audit Data Generation is included in the PP/ST:

- Decisions to DSCARD, BYPASS, PROTECT network packets processed by the TOE
- Failure to establish an IPsec SA
- IPsec SA establishment
- IPsec SA termination
- Negotiation "down" from IKEv2 to an IKEv1 exchange

FCS_IPSEC_EXT.1 IPsec Protocol

Hierarchical to: No other components

Dependencies: FCS_CKM.1 Cryptographic Key Generation,
FCS_CKM.2 Cryptographic Key Establishment,
FCS_COP.1(a) Cryptographic Operation (Signature Verification),
FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption),
FCS_RBG_EXT.1 Random Bit Generation,
FIA_X509_EXT.1 X.509 Certificate Validation

FCS_IPSEC_EXT.1.1 The TSF shall implement the IPsec architecture as specified in RFC 4301.

FCS_IPSEC_EXT.1.2 The TSF shall have a nominal, final entry in the SPD that matches anything that is otherwise unmatched, and discards it.

FCS_IPSEC_EXT.1.3 The TSF shall implement transport mode and [selection: tunnel mode, no other mode].

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms AES-CBC-128, AES-CBC-256 (both specified by RFC 3602) and [selection: AES-GCM-128 (specified in RFC 4106), AES-GCM-256 (specified in RFC 4106), no other algorithms] together with a Secure Hash Algorithm (SHA)-based HMAC.

FCS_IPSEC_EXT.1.5 The TSF shall implement the protocol: [selection:

- IKEv1, using Main Mode for Phase 1 exchanges, as defined in RFCs 2407, 2408, 2409, RFC 4109, [selection: no other RFCs for extended sequence numbers, RFC 4304 for extended sequence numbers], and [selection: no other RFCs for hash functions, RFC 4868 for hash functions];

- IKEv2 as defined in RFC 5996 and [selection: with no support for NAT traversal, with mandatory support for NAT traversal as specified in RFC 5996, section 2.23)], and [selection: no other RFCs for hash functions, RFC 4868 for hash functions]

].

FCS_IPSEC_EXT.1.6 The TSF shall ensure the encrypted payload in the [selection: IKEv1, IKEv2] protocol uses the cryptographic algorithms AES-CBC-128, AES-CBC-256 as specified in RFC 3602 and [selection: AES-GCM-128, AES-GCM-256 as specified in RFC 5282, no other algorithm].

FCS_IPSEC_EXT.1.7 The TSF shall ensure that [selection:

- IKEv1 Phase 1 SA lifetimes can be configured by an administrator based on [selection:
    - number of bytes;
    - length of time, where the time values can configured within [*assignment: integer range including 24*] hours;

  ];

- IKEv2 SA lifetimes can be configured by an administrator based on [selection:
    - number of bytes;
    - length of time, where the time values can configured within [*assignment: integer range including 24*] hours

  ]

].

FCS_IPSEC_EXT.1.8 The TSF shall ensure that [selection:

- IKEv1 Phase 2 SA lifetimes can be configured by an administrator based on [selection:
    - number of bytes;
    - length of time, where the time values can be configured within [*assignment: integer range including 8*] hours;

  ];

- IKEv2 Child SA lifetimes can be configured by an administrator based on [selection:
    - number of bytes;
    - length of time, where the time values can be configured within [*assignment: integer range including 8*] hours;

  ]

].

FCS_IPSEC_EXT.1.9 The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange (“x” in g^x mod p) using the random bit generator specified in

FCS_RBG_EXT.1, and having a length of at least [*assignment: (one or more) number(s) of bits that is at least twice the security strength of the negotiated Diffie-Hellman group*] bits.

FCS_IPSEC_EXT.1.10 The TSF shall generate nonces used in [selection: IKEv1, IKEv2] exchanges of length [selection:

- [*assignment: security strength associated with the negotiated Diffie-Hellman group*];
- at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash

].

FCS_IPSEC_EXT.1.11 The TSF shall ensure that all IKE protocols implement DH Groups 14 (2048-bit MODP), and [selection: 19 (256-bit Random ECP), 5 (1536-bit MODP), 24 (2048-bit MODP with 256-bit POS), 20 (384-bit Random ECP), no other DH groups].

FCS_IPSEC_EXT.1.12 The TSF shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [selection: IKEv1 Phase 1, IKEv2 IKE_SA] connection is greater than or equal to the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [selection: IKEv1 Phase 2, IKEv2 CHILD_SA] connection.

FCS_IPSEC_EXT.1.13 The TSF shall ensure that all IKE protocols perform peer authentication using [selection: RSA, ECDSA] that use X.509v3 certificates that conform to RFC 4945 and [selection: pre-shared keys, no other method].

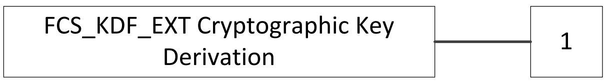
FCS_IPSEC_EXT.1.14 The TSF shall only establish a trusted channel to peers with valid certificates.

FCS_KDF_EXT Cryptographic Key Derivation

Family Behavior

This family specifies the means by which an intermediate key is derived from a specified set of submasks.

Component Leveling



FCS_KDF_EXT.1, Cryptographic Key Derivation, requires the TSF to derive intermediate keys from submasks using the specified hash functions.

Management: FCS_KDF_EXT.1

No specific management functions are identified.

Audit: FCS_KDF_EXT.1

There are no auditable events foreseen.

FCS_KDF_EXT.1 Cryptographic Key Derivation

Hierarchical to: No other components

Dependencies: FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm), FCS_RBG_EXT.1 Random Bit Generation

FCS_KDF_EXT.1.1 The TSF shall accept [selection: a RNG generated submask as specified in FCS_RBG_EXT.1, a conditioned password submask, imported submask] to derive an intermediate key, as defined in [selection:

- NIST SP 800-108 [selection: KDF in Counter Mode, KDF in Feedback Mode, KDF in Double-Pipeline Iteration Mode],
- NIST SP 800-132],

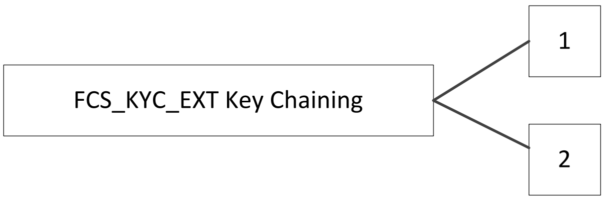
using the keyed-hash functions specified in FCS_COP.1(c), such that the output is at least of equivalent security strength (in number of bits) to the BEV.

FCS_KYC_EXT Key Chaining

Family Behavior

This family provides the specification to be used for using multiple layers of encryption keys to ultimately secure the protected data encrypted on the drive.

Component Leveling



FCS_KYC_EXT.1, Key Chaining (Initiator), requires the TSF to maintain a key chain for a BEV that is provided to a component external to the TOE.

FCS_KYC_EXT.2, Key Chaining (Recipient), requires the TSF to be able to accept a BEV that is then chained to a DEK used by the TSF through some method.

Note that this cPP-Module does not include FCS_KYC_EXT.2; it is only included here to provide a complete definition of the FCS_KYC_EXT family.

Management: FCS_KYC_EXT.1

No specific management functions are identified.

Audit: FCS_KYC_EXT.1

There are no auditable events foreseen.

Management: FCS_KYC_EXT.2

No specific management functions are identified.

Audit: FCS_KYC_EXT.2

There are no auditable events foreseen.

FCS_KYC_EXT.1 Key Chaining (Initiator)

Hierarchical to: No other components

Dependencies: FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys),
FCS_CKM.1(b) Cryptographic Operation (Symmetric Keys),
FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
FCS_COP.1(e) Cryptographic Operation (Key Transport),
FCS_COP.1(g) Cryptographic Operation (Key Encryption),
FCS_SMC_EXT.1 Submask Combining,
FCS_VAL_EXT.1 Validation

FCS_KYC_EXT.1.1 The TSF shall maintain a key chain of: [selection:

- one, using a submask as the BEV;
- intermediate keys generated by the TSF using the following method(s): [selection:
  - asymmetric key generation as specified in FCS_CKM.1(a),
  - symmetric key generation as specified in FCS_CKM.1(b)];
- intermediate keys originating from one or more submask(s) to the BEV using the following method(s): [selection:
  - key derivation as specified in FCS_KDF_EXT.1,
  - key wrapping as specified in FCS_COP.1(d),
  - key combining as specified in FCS_SMC_EXT.1,
  - key transport as specified in FCS_COP.1(e),
  - key encryption as specified in FCS_COP.1(g)]]

while maintaining an effective strength of [selection: 128 bits, 256 bits] for symmetric keys and an effective strength of [selecton: not applicable, 112 bits, 128 bits, 192 bits, 256 bits] for asymmetric keys.

FCS_KYC_EXT.1.2 The TSF shall provide a [selection: 128 bit, 256 bit] BEV to [*assignment: one or more external entities*] [selection: only after the TSF has successfully performed the validation process as specified in FCS_VAL_EXT.1, without validation taking place].

Application Note: *Key Chaining is the method of using multiple layers of encryption keys to ultimately secure the BEV. The number of intermediate keys will vary – from one (e.g., taking the conditioned password authorization factor and directly using it as the BEV) to many. This applies to all keys that contribute to the ultimate wrapping or derivation of the BEV; including those in areas of protected storage (e.g. TPM stored keys, comparison values).*

FCS_KYC_EXT.2 Key Chaining (Recipient)

Hierarchical to: No other components

Dependencies: No other components

FCS_KYC_EXT.2.1 The TSF shall accept a BEV of [selection: 128 bits, 256 bits] from [*assignment: one or more external entities*].

FCS_KYC_EXT.2.2 The TSF shall maintain a chain of intermediary keys originating from the BEV to the DEK using the following method(s): [selection:

- asymmetric key generation as specified in FCS_CKM.1(a)
- symmetric key generation as specified in FCS_CKM.1(b)
- key derivation as specified in FCS_KDF_EXT.1,
- key wrapping as specified in FCS_COP.1(d),
- key transport as specified in FCS_COP.1(e),
- key encryption as specified in FCS_COP.1(g)]

while maintaining an effective strength of [selection: 128 bits, 256 bits].

Application Note: *Key Chaining is the method of using multiple layers of encryption keys to ultimately secure the protected data encrypted on the drive. The number of intermediate keys will vary – from one (e.g., using the BEV as a key encrypting key (KEK)) to many. This applies to all keys that contribute to the ultimate wrapping or derivation of the DEK; including those in areas of protected storage (e.g. TPM stored keys, comparison values).*

FCS_PCC_EXT Cryptographic Password Construction and Conditioning

Family Behavior

This family ensures that passwords used to produce the BEV are robust (in terms of their composition) and are conditioned to provide an appropriate-length bit string.

Component Leveling

FCS_PCC_EXT.1, Cryptographic Password Construction and Conditioning, requires the TSF to accept passwords of a certain composition and condition them appropriately.

Management: FCS_PCC_EXT.1

No specific management functions are identified.

Audit: FCS_PCC_EXT.1

There are no auditable events foreseen.

FCS_PCC_EXT.1 Cryptographic Password Construction and Conditioning

Hierarchical to: No other components

Dependencies: FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

FCS_PCC_EXT.1.1 A password used by the TSF to generate a password authorization factor shall enable up to [*assignment: positive integer of 64 or more*] characters in the set of {upper case characters, lower case characters, numbers, and [*assignment: other supported special characters*]} and shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm HMAC-[selection: SHA-256, SHA-512], with [*assignment: positive integer of 1000 or more*] iterations, and output cryptographic key sizes [selection: 128 bits, 256 bits] that meet the following: [*assignment: PBKDF recommendation or specification*].

FCS_RBG_EXT Random Bit Generation

Family Behavior

Components in this family address the requirements for random bit/number generation. This is a new family defined for the FCS class.

Component Leveling

FCS_RBG_EXT.1, Random Bit Generation, requires random bit generation to be performed in accordance with selected standards and seeded by an entropy source.

Management: FCS_RBG_EXT.1

No specific management functions are identified.

Audit: FCS_RBG_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of the randomization process

FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

Hierarchical to: No other components

Dependencies: FCS_COP.1(b) Cryptographic Operation (Hash Algorithm), FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm)

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with [selection: ISO/IEC 18031:2011, [*assignment: other RBG standards*]] 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 that accumulates entropy from [selection: [*assignment: number of software-based sources*] software-based noise source(s), [*assignment: number of hardware-based sources*] hardware-based noise source(s)] with a minimum of [selection: 128 bits, 256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions", of the keys and hashes 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). The ST author will select the function used, and include the specific underlying cryptographic primitives used in the requirement. While any of the identified hash functions (SHA-256, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed.*

FCS_SMC_EXT Submask Combining

Family Behavior

This family specifies the means by which submasks are combined, if the TOE supports more than one submask being used to derive or protect the BEV.

Component Leveling

FCS_SMC_EXT.1, Submask Combining, requires the TSF to combine the submasks in a predictable fashion.

Management: FCS_SMC_EXT.1

No specific management functions are identified.

Audit: FCS_SMC_EXT.1

There are no auditable events foreseen.

FCS_SMC_EXT.1 Submask Combining

Hierarchical to: No other components

Dependencies: FCS_COP.1(b) Cryptographic Operation (Hash Algorithm)

FCS_SMC_EXT.1.1 The TSF shall combine submasks using the following method [selection: exclusive OR (XOR), SHA-256, SHA-512] to generate an [*assignment: types of keys*].

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

Family Behavior

This family ensures that salts, nonces, and IVs are well formed.

Component Leveling

FCS_SNI_EXT.1, Cryptographic Operation (Salt, Nonce, and Initialization Vector Generation), requires the generation of salts, nonces, and IVs to be used by the cryptographic components of the TOE to be performed in the specified manner.

Management: FCS_SNI_EXT.1

No specific management functions are identified.

Audit: FCS_SNI_EXT.1

There are no auditable events foreseen.

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

Hierarchical to: No other components

Dependencies: FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

FCS_SNI_EXT.1.1 The TSF shall only use salts that are generated by a [selection: DRBG as specified in FCS_RBG_EXT.1, DRBG provided by the host platform].

FCS_SNI_EXT.1.2 The TSF shall only use unique nonces, with a minimum size of [*assignment: number of bits*] bits.

FCS_SNI_EXT.1.3 The TSF shall create IVs in the following manner:

- CBC: IVs shall be non-repeating,
- CCM: Nonce shall be non-repeating,
- XTS: No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer,
- GCM: IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key.

FCS_SSHC_EXT SSH Client Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using SSH when the TSF is the sender (client) for the communications.

Component Leveling

FCS_SSHC_EXT.1, SSH Client Protocol, requires the TSF to implement SSH as a client.

Management: FCS_SSHC_EXT.1

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

- Configuration of SSH authentication method
- Configuration of SSH encryption, integrity, and key exchange algorithms

Audit: FCS_SSHC_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of SSH session establishment
- SSH session establishment
- SSH session termination

FCS_SSHC_EXT.1 SSH Client Protocol

Hierarchical to: No other components

Dependencies: FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys),
FCS_CKM.2 Cryptographic Key Establishment,
FCS_COP.1(a) Cryptographic Operation (Signature Verification),
FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption),

FCS_SSHC_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and [selection: 5647, 5656, 6187, 6668, no other RFCs].

FCS_SSHC_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based.

FCS_SSHC_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*assignment: number of bytes*] bytes in an SSH transport connection are dropped.

FCS_SSHC_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-cbc, aes256-cbc, [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other algorithms].

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH transport implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHC_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [selection: hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512] and [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other MAC algorithms] as its data integrity MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHC_EXT.1.7 The TSF shall ensure that [selection: diffie-hellman-group14-sha1, ecdh-sha2-nistp256] and [selection: ecdh-sha2-nistp384, ecdh-sha2-nistp521, no other methods] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHC_EXT.1.8 The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

FCS_SSHC_EXT.1.9 The TSF shall ensure that the SSH client authenticates the identity of the SSH server using a local database associating each host name with its corresponding public key or [selection: a list of trusted certification authorities, no other methods] as described in RFC 4251 section 4.1.

FCS_SSHS_EXT SSH Server Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using SSH when the TSF is the recipient (server) for the communications.

Component Leveling

FCS_SSHS_EXT.1, SSH Server Protocol, requires the TSF to implement SSH as a server.

Management: FCS_SSHS_EXT.1

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

- Configuration of SSH authentication method
- Configuration of SSH encryption, integrity, and key exchange algorithms

Audit: FCS_SSHS_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of SSH session establishment
- SSH session establishment
- SSH session termination

FCS_SSHS_EXT.1 SSH Server Protocol

Hierarchical to: No other components

Dependencies: FCS_CKM.1(a) Cryptographic Key Generation (Asymmetric Keys), FCS_CKM.2 Cryptographic Key Establishment, FCS_COP.1(a) Cryptographic Operation (Signature Verification), FCS_COP.1(b) Cryptographic Operation (Hash Algorithm), FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption)

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and [selection: 5647, 5656, 6187, 6668, no other RFCs].

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based.

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*assignment: number of bytes*] bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: *aes128-cbc, aes256-cbc*, [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other algorithms].

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH transport implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [selection: hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512] and [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other MAC algorithms] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7 The TSF shall ensure that [selection: diffie-hellman-group14-sha1, ecdh-sha2-nistp256] and [selection: ecdh-sha2-nistp384, ecdh-sha2-nistp521, no other methods] are the only allowed key exchange methods used for the SSH protocol.

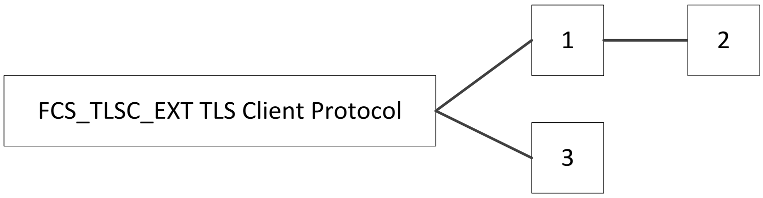
FCS_SSHS_EXT.1.8 The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

FCS_TLSC_EXT TLS Client Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using TLS when the TSF is the sender (client) for the communications.

Component Leveling



FCS_TLSC_EXT.1, TLS Client Protocol, requires the TSF to implement TLS as a client.

FCS_TLSC_EXT.2, TLS Client Protocol with Authentication, requires the TSF to implement mutual authentication in addition to the requirements of FCS_TLSC_EXT.1.

FCS_TLSC_EXT.3, TLS Client Handshake Message Exchange, defines the method by which the TSF performs the TLS handshake message exchange when PSK ciphersuites are used.

Note that this cPP-Module does not include FCS_TLSC_EXT.2; it is only included here to provide a complete definition of the FCS_TLSC_EXT family.

Management: FCS_TLSC_EXT.1

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

- Configuration of supported TLS version and ciphersuite(s)
- Specification of supported elliptic curves to be presented in Client Hello

Audit: FCS_TLSC_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of TLS session establishment
- TLS session establishment
- TLS session termination

Management: FCS_TLSC_EXT.2

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

- Configuration of supported TLS version and ciphersuite(s)
- Specification of supported elliptic curves to be presented in Client Hello

Audit: FCS_TLSC_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of TLS session establishment
- TLS session establishment
- TLS session termination

Management: FCS_TLSC_EXT.3

No specific management functions are identified.

Audit: FCS_TLSC_EXT.3

There are no auditable events foreseen.

FCS_TLSC_EXT.1 TLS Client Protocol

Hierarchical to: No other components

Dependencies: FCS_COP.1(a) Cryptographic Operation (Signature Verification), FCS_COP.1(b) Cryptographic Operation (Hash Algorithm), FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption), FCS_RBG_EXT.1 Random Bit Generation, FIA_X509_EXT.1 X.509 Certificate Validation, FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSC_EXT.1.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3 The TSF shall only establish a trusted channel if the peer certificate is valid.

FCS_TLSC_EXT.1.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1, or none] and no other curves.

FCS_TLSC_EXT.1.5 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

FCS_TLSC_EXT.2 TLS Client Protocol with Authentication

Hierarchical to: FCS_TLSC_EXT.1 TLS Client Protocol

Dependencies: FCS_COP.1(a) Cryptographic Operation (Signature Verification),
FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption),
FCS_RBG_EXT.1 Random Bit Generation,
FIA_X509_EXT.1 X.509 Certificate Validation,
FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSC_EXT.2.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

FCS_TLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.2.3 The TSF shall only establish a trusted channel if the peer certificate is valid.

FCS_TLSC_EXT.2.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1, or none] and no other curves.

FCS_TLSC_EXT.2.5 The TSF shall support mutual authentication using X.509v3 certificates.

FCS_TLSC_EXT.3 TLS Client Handshake Message Exchange

Hierarchical to: No other components

Dependencies: FCS_TLSC_EXT.1 TLS Client Protocol

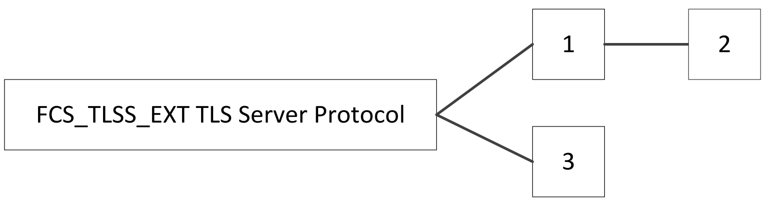
FCS_TLSC_EXT.3.1 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

FCS_TLSS_EXT TLS Server Protocol

Family Behavior

Components in this family describe the requirements for protecting remote communications using TLS when the TSF is the recipient (server) for the communications.

Component Leveling



FCS_TLSS_EXT.1, TLS Server Protocol, requires the TSF to implement TLS as a client.

FCS_TLSS_EXT.2, TLS Server Protocol with Authentication, requires the TSF to implement mutual authentication in addition to the requirements of FCS_TLSC_EXT.1.

FCS_TLSS_EXT.3, TLS Server Handshake Message Exchange, defines the method by which the TSF performs the TLS handshake message exchange when PSK ciphersuites are used.

Note that this cPP-Module does not include FCS_TLSS_EXT.2; it is only included here to provide a complete definition of the FCS_TLSS_EXT family.

Management: FCS_TLSS_EXT.1

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

- Configuration of supported TLS version and ciphersuite(s)
- Specification of key establishment parameters

Audit: FCS_TLSS_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of TLS session establishment
- TLS session establishment
- TLS session termination

Management: FCS_TLSS_EXT.2

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

- Configuration of supported TLS version and ciphersuite(s)
- Specification of key establishment parameters

Audit: FCS_TLSS_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Failure of TLS session establishment
- TLS session establishment
- TLS session termination

Management: FCS_TLSC_EXT.3

No specific management functions are identified.

Audit: FCS_TLSC_EXT.3

There are no auditable events foreseen.

FCS_TLSS_EXT.1 TLS Server Protocol

Hierarchical to: No other components

Dependencies: FCS_COP.1(a) Cryptographic Operation (Signature Verification),
FCS_COP.1(b) Cryptographic Operation (Hash Algorithm),
FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption),
FCS_RBG_EXT.1 Random Bit Generation,
FIA_X509_EXT.1 X.509 Certificate Validation,
FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSS_EXT.1.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting SSL 1.0, SSL 2.0, SSL 3.0, TLS 1.0, and [selection: TLS 1.1, TLS 1.2, none].

FCS_TLSS_EXT.1.3 The TSF shall generate key establishment parameters using RSA with key size 2048 bits and [selection: 3072 bits, 4096 bits, no other size] and [selection: over NIST curves [selection: secp256r1, secp384r1] and no other curves; Diffie-Hellman parameters of size 2048 bits and [selection: 3072 bits, no other size]; no other].

FCS_TLSS_EXT.1.4 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

FCS_TLSS_EXT.2 TLS Server Protocol with Authentication

Hierarchical to:	FCS_TLSS_EXT.1 TLS Server Protocol
Dependencies:	FCS_COP.1(a) Cryptographic Operation (Signature Verification), FCS_COP.1(b) Cryptographic Operation (Hash Algorithm), FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption), FCS_RBG_EXT.1 Random Bit Generation, FIA_X509_EXT.1 X.509 Certificate Validation,

FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSS_EXT.2.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] supporting the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 as defined in RFC 5487
- TLS_RSA_PSK_WITH_AES_256_GCM_SHA384 as defined in RFC 5487
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 as defined in RFC 5489
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 as defined in RFC 5489

].

FCS_TLSS_EXT.2.2 The TSF shall deny connections from clients requesting SSL 1.0, SSL 2.0, SSL 3.0, TLS 1.0, and [selection: TLS 1.1, TLS 1.2, none].

FCS_TLSS_EXT.2.3 The TSF shall generate key establishment parameters using RSA with key size 2048 bits and [selection: 3072 bits, 4096 bits, no other size] and [selection: over NIST curves [selection: secp256r1, secp384r1] and no other curves; Diffie-Hellman parameters of size 2048 bits and [selection: 3072 bits, no other size]; no other].

FCS_TLSS_EXT.2.4 The TSF shall support mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.5 The TSF shall not establish a trusted channel if the peer certificate is invalid.

FCS_TLSS_EXT.2.6 The TSF shall not establish a trusted channel if the distinguished name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match the expected identifier for the peer.

FCS_TLSS_EXT.3 TLS Server Handshake Message Exchange

Hierarchical to: No other components

Dependencies: FCS_TLSS_EXT.1 TLS Server Protocol

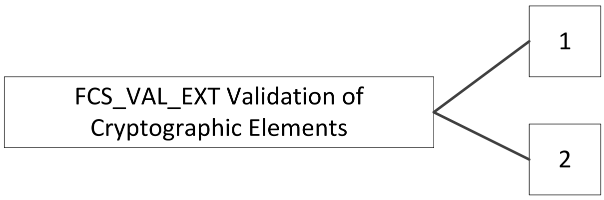
FCS_TLSS_EXT.3.1 The TSF operating within the intra-TOE client/server communication channel shall [selection: use full TLS handshake message exchange, use reduced TLS handshake message exchange].

FCS_VAL_EXT Validation of Cryptographic Elements

Family Behavior

This family specifies the means by which submasks and/or BEVs are determined to be valid prior to their use.

Component Leveling



FCS_VAL_EXT.1, Validation, requires the TSF to validate submasks and BEVs by one or more of the specified methods.

FCS_VAL_EXT.2, User Validation, requires the TSF to validate the legitimacy of a user's request before providing cryptographic data to the user.

Management: FCS_VAL_EXT.1

No specific management functions are identified.

Audit: FCS_VAL_EXT.1

There are no auditable events foreseen.

Management: FCS_VAL_EXT.2

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

- Specification of the validation method used
- Configuration of number of failed validation attempts that will be accepted by the TSF
- Action taken by the TSF in the event an unacceptable number of failed validation attempts are made

Audit: FCS_VAL_EXT.2

There are no auditable events foreseen.

FCS_VAL_EXT.1 Validation

Hierarchical to: No other components

Dependencies: FCS_COP.1(b) Cryptographic Operation (Hash Algorithm), FCS_COP.1(c) Cryptographic Operation (Keyed Hash Algorithm), FCS_COP.1(d) Cryptographic Operation (Key Wrapping), FCS_COP.1(f) Cryptographic Operation (AES Data Encryption/Decryption)

FCS_VAL_EXT.1.1 The TSF shall perform validation of the [selection: submask, intermediate key, BEV] using the following method(s): [selection:

- key wrap as specified in FCS_COP.1(d);
- hash the [selection: submask, intermediate key, BEV] as specified in [selection: FCS_COP.1(b), FCS_COP.1(c)] and compare it to a stored hashed [selection: submask, intermediate key, BEV];
- decrypt a known value using the [selection: submask, intermediate key, BEV] as specified in FCS_COP.1(f) and compare it against a stored known value].

FCS_VAL_EXT.1.2 The TSF shall require validation of the [selection: submask, intermediate key, BEV] prior to [*assignment: activity requiring validation*].

FCS_VAL_EXT.1.3 The TSF shall [selection: [*assignment: key sanitization activity*] upon a configurable number of consecutive failed validation attempts, institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period, block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts, require power cycle of or reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

FCS_VAL_EXT.2 User Validation

FCS_VAL_EXT.2.1 The TSF shall perform validation of the [user] by receiving assertion of the user's validity from: [*assignment: Operational Environment component responsible for user authentication*].

FCS_VAL_EXT.2.2 The TSF shall require validation of the user prior to [*assignment: cryptographic operation or transmission of cryptographic data*].

FCS_VAL_EXT.2.3 The TSF shall [selection: [*assignment: key sanitization activity*] upon receiving a configurable number of consecutive failed validation attempts from the Operational Environment; institute a delay such that only [*assignment: ST author specified number of attempts*] can be made within a 24 hour period; block validation after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts; require power cycle of or reset the TOE after [*assignment: ST author specified number of attempts*] of consecutive failed validation attempts].

FIA_CHR_EXT Challenge/Response Recovery Credential

Family Behavior

This family defines characteristics of a challenge/response recovery credential if one is supported by the TOE.

Component Leveling

FIA_CHR_EXT.1, Challenge/Response Recovery Credential, requires the TSF to define the circumstances under which a challenge/response credential can be generated and used.

Management: FIA_CHR_EXT.1

No specific management functions are identified.

Audit: FIA_CHR_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Generation of response

FIA_CHR_EXT.1 Challenge/Response Recovery Credential

Hierarchical to: No other components

Dependencies: FIA_REC_EXT.1 Support for Recovery Credentials

FIA_CHR_EXT.1.1 The TSF shall only generate a response if it is able to access recovery information for [selection: the user requesting the recovery, the device for which the recovery was requested].

FIA_CHR_EXT.1.2 The response shall only work on the system upon which the challenge was generated.

FIA_CHR_EXT.1.3 The response shall only be used during the same session in which the request was generated.

FIA_CHR_EXT.1.4 The TSF shall generate an ephemeral response that has at least as many potential values as a corresponding password or PIN.

FIA_CHR_EXT.1.5 The TSF shall allow a maximum of [*assignment: integer value*] of response entry attempts per boot cycle.

FIA_CHR_EXT.1.6 The TSF shall [selection:

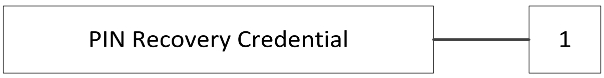
- perform a key sanitization of the DEK upon [*assignment: ST author specified number or configurable range of attempts*] consecutive failed validation attempts,
- institute a delay such that only [*assignment: ST author specified number or configurable range of attempts*] validation attempts can be made within a 24 hour period,
- block validation after [*assignment: ST author specified number or configurable range of attempts*] of consecutive failed validation attempts,
- terminate the session after [*assignment: ST author specified number or configurable range of attempts*] consecutive failed validation attempts].

FIA_PIN_EXT PIN Recovery Credential

Family Behavior

This family defines characteristics of a PIN recovery credential if one is supported by the TOE.

Component Leveling



FIA_PIN_EXT.1, PIN Recovery Credential, requires the TSF to pre-populate the PIN recovery credential and limit it to a single use on a single system.

Management: FIA_PIN_EXT.1

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

- Setting PIN value

Audit: FIA_PIN_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Setting PIN value
- Use of PIN value for authentication

FIA_PIN_EXT.1 PIN Recovery Credential

Hierarchical to: No other components

Dependencies: FIA_REC_EXT.1 Support for Recovery Credentials

FIA_PIN_EXT.1.1 The TSF shall pre-populate the recovery PIN on the Management Server.

FIA_PIN_EXT.1.2 The recovery key chain accessed by the recovery PIN shall only work on the system within which the drive or set of drives to be recovered resides.

FIA_PIN_EXT.1.3 The TSF shall not permit the PIN to be used more than once.

FIA_REC_EXT Support for Recovery Credentials

Family Behavior

This family defines the ability of the TOE to use a recovery credential as an alternative authentication mechanism in the event of a lost or forgotten authentication factor.

Component Leveling

FIA_REC_EXT.1, Support for Recovery Credentials, defines the ability of the TOE to use (or not use) a recovery credential and allows the ST author to specify the type(s) of credentials that are supported.

Management: FIA_REC_EXT.1

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

- Enabling and disabling of recovery credential support
- Specification of the recovery credential type(s) to be used

Audit: FIA_REC_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Enabling and disabling of recovery credential support

FIA_REC_EXT.1 Support for Recovery Credentials

Hierarchical to: No other components

Dependencies: No dependencies

FIA_REC_EXT.1.1 The TSF shall support the following recovery credentials: [selection: challenge/response, PIN].

FIA_REC_EXT.1.2 The TSF shall provide the ability to enable and disable the use of recovery credentials.

FIA_X509_EXT Authentication Using X.509 Certificates

This family defines the behavior, management, and use of X.509 certificates for functions to be performed by the TSF. Components in this family require validation of certificates according to a specified set of rules, use of certificates for authentication for protocols and integrity verification, and the generation of certificate requests.

Component Leveling

FIA_X509_EXT.1, X509 Certificate Validation, requires the TSF to check and validate certificates in accordance with the RFCs and rules specified in the component.

FIA_X509_EXT.2, X509 Certificate Authentication, requires the TSF to use certificates to authenticate peers in protocols that support certificates, as well as for integrity verification and potentially other functions that require certificates.

FIA_X509_EXT.3, X509 Certificate Requests, requires the TSF to be able to generate Certificate Request Messages and validate responses.

Management: FIA_X509_EXT.1

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

- Import and removal of X.509v3 certificates
- Approval of import and removal of X.509v3 certificates
- Initiation of certificate validation requests

Audit: FIA_X509_EXT.1

There are no auditable events foreseen.

Management: FIA_X509_EXT.2

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

- Import and removal of X.509v3 certificates
- Approval of import and removal of X.509v3 certificates
- Initiation of certificate validation requests

Audit: FIA_X509_EXT.2

There are no auditable events foreseen.

Management: FIA_X509_EXT.3

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

- Import and removal of X.509v3 certificates
- Approval of import and removal of X.509v3 certificates
- Initiation of certificate validation requests

Audit: FIA_X509_EXT.3

There are no auditable events foreseen.

FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1 The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The TSF shall validate the revocation status of the certificate using [selection: the Online Certificate Status Protocol (OCSP) as specified in RFC 2560, a Certificate Revocation List (CRL) as specified in RFC 5759].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
    - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

FIA_X509_EXT.1.2 The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [selection: IPsec, TLS, HTTPS, SSH], and [no additional uses].

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [selection: allow the administrator to choose whether to accept the certificate in these cases, accept the certificate, not accept the certificate].

FIA_X509_EXT.3 X.509 Certificate Requests

FIA_X509_EXT.3.1 The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [selection: device-specific information, Common Name, Organization, Organizational Unit, Country].

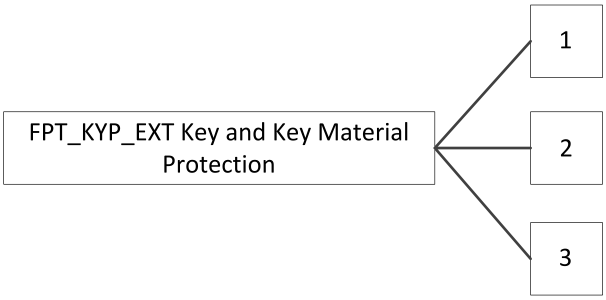
FIA_X509_EXT.3.2 The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

FPT_KYP_EXT Key and Key Material Protection

Family Behavior

This family requires that key and key material be protected if and when written to non-volatile storage.

Component Leveling



FPT_KYP_EXT.1, Protection of Key and Key Material, requires the TSF to ensure that no plaintext key or key material are written to non-volatile storage.

FPT_KYP_EXT.2, Storage of Protected Key and Key Material, requires the TSF to specify the non-volatile storage location in which encrypted key and key material is stored.

FPT_KYP_EXT.3, Attribution of Protected Key and Key Material, requires the TSF to maintain an association between encrypted key and key material and the subjects that are authorized to decrypt and/or use the data.

Management: FPT_KYP_EXT.1

No specific management functions are identified.

Audit: FPT_KYP_EXT.1

There are no auditable events foreseen.

Management: FPT_KYP_EXT.2

No specific management functions are identified.

Audit: FPT_KYP_EXT.2

There are no auditable events foreseen.

Management: FPT_KYP_EXT.3

No specific management functions are identified.

Audit: FPT_KYP_EXT.3

There are no auditable events foreseen.

FPT_KYP_EXT.1 Protection of Key and Key Material

Hierarchical to: No other components

Dependencies: FCS_COP.1(d) Cryptographic Operation (Key Wrapping),
FCS_COP.1(e) Cryptographic Operation (Key Transport),
FCS_COP.1(g) Cryptographic Operation (Key Encryption),
FCS_KYC_EXT.1 Key Chaining (Initiator),
FCS_KYC_EXT.2 Key Chaining (Recipient),
FCS_SMC_EXT.1 Submask Combining

FPT_ KYP _EXT.1.1 The TSF shall only store keys in non-volatile memory when wrapped, as specified in FCS_COP.1(d) or encrypted, as specified in FCS_COP.1(g) or FCS_COP.1(e), unless the key meets any one of following criteria [selection:

- The plaintext key is not part of the key chain as specified in [selection: FCS_KYC_EXT.1, FCS_KYC_EXT.2].
- The plaintext key will no longer provide access to the encrypted data after initial provisioning.
- The plaintext key is a key split that is combined as specified in FCS_SMC_EXT.1, and the other half of the key split is [selection: wrapped as specified in FCS_COP.1(d), encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e), derived and not stored in non-volatile memory].
- The plaintext key is stored on an external storage device for use as an authorization factor.
- The plaintext key is [selection: used to wrap a key as specified in FCS_COP.1(d), encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)] that is already [selection: wrapped as specified in FCS_COP.1(d), encrypted as specified in FCS_COP.1(g) or FCS_COP.1(e)]].

FPT_KYP_EXT.2 Storage of Protected Key and Key Material

Hierarchical to: No other components

Dependencies: FPT_KYP_EXT.1 Protection of Key and Key Material

FPT_KYP_EXT.2.1 The TSF shall only store protected key and key material [selection: within the TSF, in a SQL database in the Operational Environment, [*assignment: other key storage location*]].

FPT_KYP_EXT.3 Attribution of Protected Key and Key Material

Hierarchical to: No other components

Dependencies: FPT_KYP_EXT.1 Protection of Key and Key Material,
FPT_ITT.1 Basic Internal TSF Data Transfer Protection or

FTP_ITC.1 Inter-TSF Trusted Channel

FPT_KYP_EXT.3.1 The TSF shall maintain an association between [*assignment: list of key and key material*] and [*assignment: subjects that are authorized to use the identified key and key material*].

FPT_KYP_EXT.3.2 The TSF shall provide the ability to register remote endpoints by [*assignment: exchange of mutually identifying information that allows for an association to be made*].

FPT_KYP_EXT.3.3 The TSF shall provide the ability to revoke the registration of remote endpoints by [*assignment: method of removing and/or exchanging information that prevents further communications between the TOE and the endpoint*].

FPT_KYP_EXT.3.4 The TSF shall transmit any secure or private cryptographic information that is transferred between the TOE and a remote endpoint in order to establish or disestablish an association using a communications channel with a security strength at least as great as the strength of the information being transmitted.

Appendix D: Entropy Documentation and Assessment

The Base-PP defines requirements for the product vendor or ST author to document the entropy source(s) used by the TOE to seed the deterministic random bit generator if the TSF includes the optional SFR FCS_RBG_EXT.1/Server. These same requirements apply to the PP-Configuration if any part of the TOE provides its own random bit generation function rather than rely on one that exists in its Operational Environment. If the TOE uses multiple different entropy sources for distinct random bit generation functions (e.g. the Management Server uses a different entropy source from the AA), each entropy source shall be described as part of the same entropy documentation but the author shall make it clear which entropy source(s) apply to each random bit generation function that the TOE provides.

Appendix E: Key Management Description

The Base-PP provides requirements for a key management description (KMD) so that the security of the key hierarchy is demonstrated to the evaluator. This cPP-Module includes the same requirement; however, a separate KMD does not need to be created. The entire PP-Configuration can be represented within the same KMD as long as the author clearly represents which aspects of the KMD are associated with each individual component of the TOE.

1 Appendix F: Glossary

Term	Meaning
Authorization Factor	A value that a user knows, has, or is (e.g. password, token, etc.) submitted to the TOE to establish that the user is in the community authorized to use the hard disk. This value is used in the derivation or decryption of the BEV and eventual decryption of the DEK. Note that these values may or may not be used to establish the particular identity of the user.
Assurance	Grounds for confidence that a TOE meets the SFRs [CC1].
Border Encryption Value	A value passed from the AA to the EE intended to link the key chains of the two components.
Key Sanitization	A method of sanitizing encrypted data by securely overwriting the key that was encrypting the data.
Data Encryption Key (DEK)	A key used to encrypt data-at-rest.
Full Drive Encryption	Refers to partitions of logical blocks of user accessible data as managed by the host system that indexes and partitions and an operating system that maps authorization to read or write data to blocks in these partitions. For the sake of this Security Program Definition (SPD) and cPP, FDE performs encryption and authorization on one partition, so defined and supported by the OS and file system jointly, under consideration. FDE products encrypt all data (with certain exceptions) on the partition of the storage device and permits access to the data only after successful authorization to the FDE solution. The exceptions include the necessity to leave a portion of the storage device (the size may vary based on implementation) unencrypted for such things as the Master Boot Record (MBR) or other AA/EE pre-authentication software. These FDE cPPs interpret the term "full drive encryption" to allow FDE solutions to leave a portion of the storage device unencrypted so long as it contains no protected data.
Intermediate Key	A key used in a point between the initial user authorization and the DEK.
Host Platform	The local hardware and software the TOE is running on, this does not include any peripheral devices (e.g. USB devices) that may be connected to the local hardware and software.
Key Chaining	The method of using multiple layers of encryption keys to protect data. A top layer key encrypts a lower layer key which encrypts the data; this method can have any number of layers.
Key Encryption Key (KEK)	A key used to encrypt other keys, such as DEKs or storage that contains keys.
Key Material	Key material is commonly known as critical security parameter (CSP) data, and also includes authorization data, nonces, and metadata.
Key Release Key (KRK)	A key used to release another key from storage, it is not used for the direct derivation or decryption of another key.
Operating System (OS)	Software which runs at the highest privilege level and can directly control hardware resources.
Non-Volatile Memory	A type of computer memory that will retain information without power.
Powered-Off State	The device has been shut down.

Term	Meaning
Protected Data	This refers to all data on the storage device with the exception of a small portion required for the TOE to function correctly. It is all space on the disk a user could write data to and includes the operating system, applications, and user data. Protected data does not include the Master Boot Record or Pre-authentication area of the drive – areas of the drive that are necessarily unencrypted.
Submask	A submask is a bit string that can be generated and stored in a number of ways.
Target of Evaluation	A set of software, firmware and/or hardware possibly accompanied by guidance. [CC1]

1 See [CC1] for other Common Criteria abbreviations and terminology.

Appendix G: Acronyms

Acronym	Meaning
AA	Authorization Acquisition
AES	Advanced Encryption Standard
BEV	Border Encryption Value
BIOS	Basic Input Output System
CBC	Cipher Block Chaining
CC	Common Criteria
CCM	Counter with CBC-Message Authentication Code
CEM	Common Evaluation Methodology
CPP	Collaborative Protection Profile
DEK	Data Encryption Key
DRBG	Deterministic Random Bit Generator
DSS	Digital Signature Standard
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
EE	Encryption Engine
EEPROM	Electrically Erasable Programmable Read-Only Memory
FIPS	Federal Information Processing Standards
FDE	Full Drive Encryption
FFC	Finite Field Cryptography
GCM	Galois Counter Mode
HMAC	Keyed-Hash Message Authentication Code
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
ITSEF	IT Security Evaluation Facility
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
IV	Initialization Vector
KEK	Key Encryption Key
KMD	Key Management Description
KRK	Key Release Key
LDAP	Lightweight Directory Access Protocol
MBR	Master Boot Record
NIST	National Institute of Standards and Technology
OS	Operating System
RBG	Random Bit Generator
RNG	Random Number Generator
RSA	Rivest Shamir Adleman Algorithm
SAR	Security Assurance Requirement
SED	Self Encrypting Drive
SHA	Secure Hash Algorithm
SFR	Security Functional Requirement
SPD	Security Problem Definition
SPI	Serial Peripheral Interface
ST	Security Target
TOE	Target of Evaluation
TPM	Trusted Platform Module
TSF	TOE Security Functionality
TSS	TOE Summary Specification
USB	Universal Serial Bus
XOR	Exclusive or
XTS	XEX (XOR Encrypt XOR) Tweakable Block Cipher with Ciphertext Stealing

Appendix H: References

National Institute of Standards and Technology (NIST) Special Publication 800-38F, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, National Institute of Standards and Technology, December 2012.

National Institute of Standards and Technology (NIST) Special Publication 800-56B, Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography, National Institute of Standards and Technology, August 2009.

National Institute of Standards and Technology (NIST) Special Publication 800-88 Revision 1, Guidelines for Media Sanitization, National Institute of Standards and Technology, December 2014.

National Institute of Standards and Technology (NIST) Special Publication 800-90A, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, National Institute of Standards and Technology, January 2012.

National Institute of Standards and Technology (NIST) Special Publication 800-132, Recommendation for Password-Based Key Derivation Part 1: Storage Applications, National Institute of Standards and Technology, December 2010.

Federal Information Processing Standard Publication (FIPS-PUB) 186-4, Digital Signature Standard (DSS), National Institute of Standards and Technology, July 2013.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 9796-2:2010 (3rd edition), Information technology — Security techniques — Digital signature schemes giving message recovery, International Organization for Standardization/International Electrotechnical Commission, 2010.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 9797-2:2011 (2nd edition), Information technology — Security techniques — Message Authentication Codes (MACs) – Part 2: Mechanisms using a dedicated hash-function, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 10116:2006 (3rd edition), Information technology — Security techniques — Modes of operation for an n-bit block cipher, International Organization for Standardization/International Electrotechnical Commission, 2006.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 10118-3:2004 (3rd edition), Information technology — Security techniques — Hash-functions – Part 3: Dedicated hash-functions, International Organization for Standardization/International Electrotechnical Commission, 2004.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 14888-3:2006 (2nd edition), Information technology — Security techniques —Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms, International Organization for Standardization/International Electrotechnical Commission, 2006.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 18031:2011 (2nd edition), Information technology — Security techniques — Random bit generation, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 18033-3:2011 (3rd edition), Information technology — Security techniques — Encryption algorithms – Part 3: Block ciphers, International Organization for Standardization/International Electrotechnical Commission, 2011.

International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 19772:2009, Information technology — Security techniques Authenticated encryption, International Organization for Standardization/International Electrotechnical Commission, 2009.