collaborative Protection Profile for Stateful Traffic Filter Firewalls

Version 2.0

6-December-2017

Acknowledgements

This collaborative Protection Profile (cPP) was developed by the Network 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 (cPP) to express the security functional requirements (SFRs) and security assurance requirements (SARs) for a Stateful Traffic Filter Firewall. The Evaluation Activities that specify the actions the evaluator performs to determine if a product satisfies the SFRs captured within this cPP are described in [SD-ND] and [SD-FW].

0.2 Scope of Document

The scope of the cPP within the development and evaluation process is described in the Common Criteria for Information Technology Security Evaluation [CC]. In particular, a cPP defines the IT security requirements of a generic type of TOE and specifies the functional and assurance security measures to be offered by that TOE to meet stated requirements [CC1, Section C.1].

0.3 Intended Readership

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

Although the cPPs and SDs may contain minor editorial errors, cPPs are recognized as living documents and the iTCs are dedicated to ongoing updates and revisions. Please report any issues to the NDFW iTC.

0.4 Related Documents

Common Criteria[1]

[CC1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model, CCMB-2012-09-001, Version 3.1 Revision 4, September 2012.

[CC2] Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, CCMB-2012-09-002, Version 3.1 Revision 4, September 2012.

[CC3] Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components, CCMB-2012-09-003, Version 3.1 Revision 4, September 2012.

[CEM] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, CCMB-2012-09-004, Version 3.1, Revision 4, September 2012.

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

Other Documents

[SD-FW] Evaluation Activities for Stateful Traffic Filter Firewalls cPP, Version 2.0, 9 October 2017

[SD-ND] **Evaluation Activities for Network Device cPP, Version 2.0, 5 May 2017**

0.5 Revision History

Version	Date	Description
2.0	6-December-2017	Released for use
1.1	21-Jul-2016	Updated draft published for public review
1.0	27-Feb-2015	Released for use
0.4	26-Jan-2015	Incorporated comments received from the CCDB review
0.3	17-Oct-2014	Draft version released to accompany CCDB review of Supporting Document.
0.2	13-Oct-2014	Internal draft in response to public review comments, for iTC review
0.1	05-Sep-2014	Draft published for Public review

Contents

Figures / Tables

1. PP Introduction

1.1 PP Reference Identification

PP Reference: collaborative Protection Profile for Stateful Traffic Filter Firewalls

PP Version: 2.0

PP Date: 6-December-2017

1.2 TOE Overview

This collaborative Protection Profile (cPP) defines requirements for the evaluation of Stateful Traffic Filter Firewalls. Such products are generally boundary protection devices, such as dedicated firewalls, routers, or perhaps even switches designed to control the flow of information between attached networks. While in some cases, firewalls implementing security features serve to segregate two distinct networks – a trusted or protected enclave and an untrusted internal or external network such as the Internet – that is only one of many possible applications. It is common for firewalls to have multiple physical network connections enabling a wide range of possible configurations and network information flow policies.

The TOE may be standalone or distributed, where a distributed TOE is one that requires multiple distinct components to operate as a logical whole in order to fulfil the requirements of this cPP (a more extensive description of distributed Stateful Traffic Filter Firewall TOEs is given in section 3).

A Virtual Stateful Traffic Filter Firewall (vTFFW) is a software implementation of firewall functionality that runs inside a virtual machine. This cPP expressly excludes evaluation of vTFFWs unless the product is able to meet all the requirements and assumptions of a physical TFFW as required in this cPP

This means:

- The virtualisation layer (or hypervisor or Virtual Machine Manager (VMM)) is considered part of the TFFW's software stack, and thus is part of the TOE and must satisfy the relevant SFRs (e.g. by treating hypervisor Administrators as Security Administrators)[2]. vTFFWs that can run on multiple VMMs must be tested on each claimed VMM unless the vendor can successfully argue equivalence.

- The physical hardware is likewise included in the TOE (as in the example included above). vTFFWs must be tested for each claimed hardware platform unless the vendor can successfully argue equivalence.

- There is only one vTFFW instance for each physical hardware platform.

[2] It may be useful to iterate the relevant SFRs in a Security Target to cover properties of the virtualisation software separately.

- There are no other guest VMs on the physical platform providing non-stateful traffic filtering firewall functionality.

1.3 TOE Use Cases

This cPP specifically addresses firewalls that perform network layer 3 and 4 stateful traffic filtering. A Stateful Traffic Filter Firewall is a device composed of hardware and software that is connected to two or more distinct networks and has an infrastructure role in the overall enterprise network.

Stateful traffic filtering is the idea that the firewall would keep track of the state of each connection through it and have the ability to drop packets that do not appear to belong to a valid flow. Information such as the TCP sequence number, ACKs, IP options are also kept by storing the metrics in dynamic state tables. Other considerations in the decision to accept, drop, or log packets are source and destination IP addresses and ports, or when the source or destination addresses are inconsistent with the configured interfaces.

Future drafts of this cPP are envisioned, which will include optional functionality (e.g., transparent mode). Future Firewall PPs will be used to specify sets of additional functionality (e.g., Application Filtering). In the context of this PP, additional features such as these are simply ignored for the purpose of evaluation except where they may have some effect of the security requirements defined herein. While many devices that will be evaluated against this PP have the capability to perform NAT or PAT, there are no requirements that specify this capability.

2. CC Conformance

As defined by the references [CC1], [CC2] and [CC3], this cPP:

- conforms to the requirements of Common Criteria v3.1, Revision 4
- is Part 2 extended, Part 3 conformant
- does not claim conformance to any other PP.

The methodology applied for the cPP evaluation is defined in [CEM]. This cPP satisfies the following Assurance Families: APE_CCL.1, APE_ECD.1, APE_INT.1, APE_OBJ.1, APE_REQ.1 and APE_SPD.1.

In order to be conformant to this cPP, a TOE must demonstrate Exact Conformance. Exact Conformance, as a subset of Strict Conformance as defined by the CC, is defined as the ST containing all of the SFRs in section 6 (these are the mandatory SFRs) of this cPP, and potentially SFRs from Appendix A (these are optional SFRs) or Appendix B (these are selection-based SFRs, some of which will be mandatory according to the selections made in other SFRs) of this cPP. While iteration is allowed, no additional requirements (from the CC parts 2 or 3, or definitions of extended components not already included in this cPP) are allowed to be included in the ST. Further, no SFRs in section 6 of this cPP are allowed to be omitted.

3. Introduction to Distributed TOEs

This cPP includes support for distributed Stateful Traffic Filter Firewall TOEs. Stateful Traffic Filter Firewalls can sometimes be composed of multiple components operating as a logical whole. Oftentimes we see this architecture when dealing with products where a centralized management console is used to provide administration to dispersed components.

There are a number of different architectures; but fundamentally, they are variations of the following model where the SFRs of this cPP can only be fulfilled if the two components are deployed and operate together.

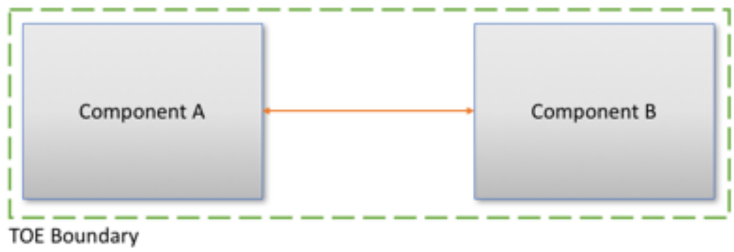


Figure 1: Generalized Distributed TOE Model

3.1 Supported Distributed TOE Use Cases

The following discussion provides guidance over the supported distributed TOE use cases in this version of the cPP.

Case 1: cPP requirements can only be fulfilled if several TOE components work together

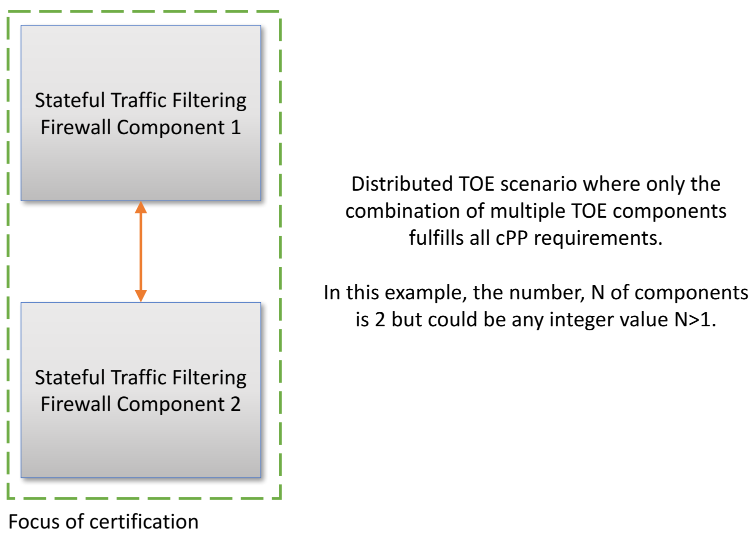


Figure 2: Basic distributed TOE use case

The first and most basic use case is where multiple interconnected Stateful Traffic Filter Firewall components need to operate together to fulfil the requirements of the cPP. To be considered a distributed TOE, a minimum of 2 interconnected components are required.

Case 2: cPP requirements can be fulfilled without Management component.

Some Stateful Traffic Filter Firewalls are designed to operate alongside a Management Component. A Stateful Traffic Filter Firewall that operates in this manner but can satisfy all SFRs of the cPP without the Management Component shall not be regarded as a distributed TOE and shall be certified according to this cPP without the Management Component

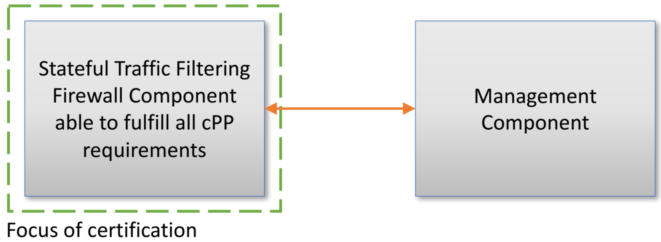


Figure 3: Non-distributed TOE use case

Alternatively, a Stateful Traffic Filter Firewall may require more than one component in order to fulfil all of the requirements of the cPP. In addition to the components required to fulfil the cPP a Management Component may also be offered for use with the TOE. However, as with the case shown in **Error! Reference source not found.** above, certification shall not include the Management Component in this case. This situation is depicted in **Error! Reference source not found.**.

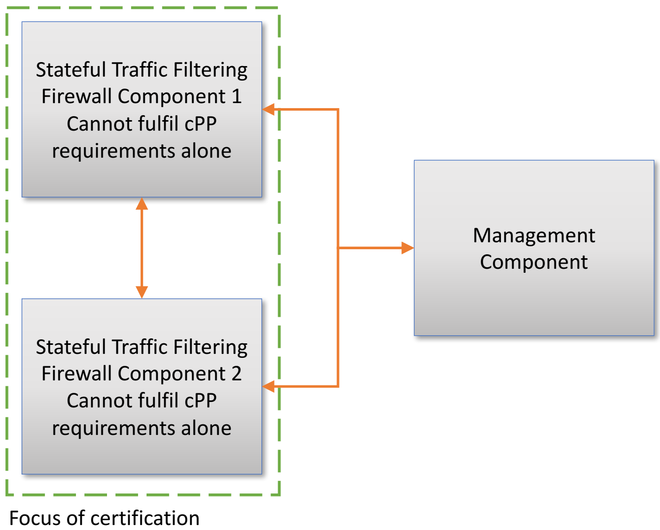


Figure 4: Distributed TOE use case with Management Component out of scope

For the cases in both **Error! Reference source not found.** and **Error! Reference source not found.**, the Management Component may be certified separately according to a different (c)PP.

Case 3: cPP requirements cannot be fulfilled without Management Component

A Stateful Traffic Filter Firewall that requires the Management Component to satisfy all SFRs of the cPP shall be considered to be a distributed TOE and be certified according to this cPP together with the Management Component.

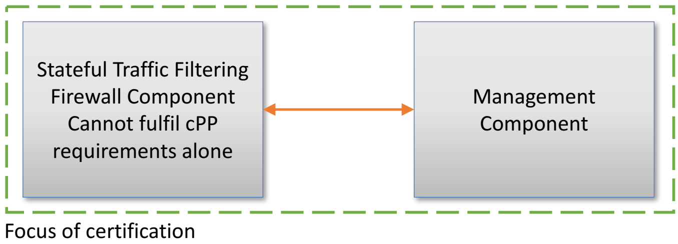


Figure 5: Management Component required to fulfil cPP requirements

A Management Component may also be considered part of the distributed TOE alongside multiple distributed Stateful Traffic Filter Firewalls if it is required to fulfil all SFRs of this cPP.

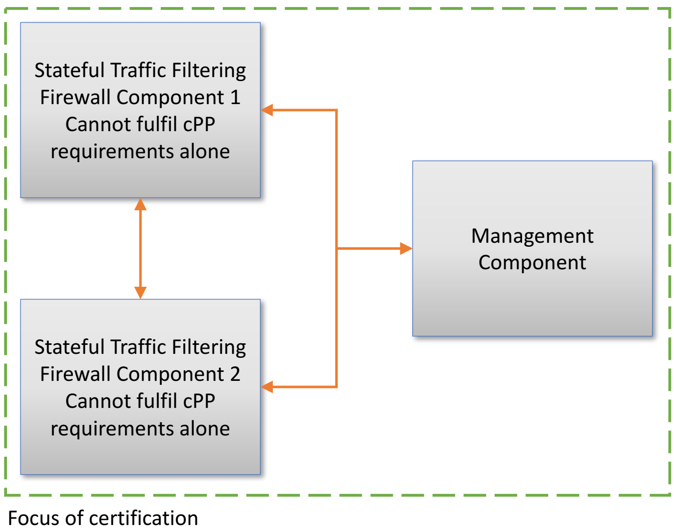


Figure 6: Distributed Stateful Traffic Filter Firewalls plus Management Component required to fulfil cPP requirements

Where several Stateful Traffic Filter Firewalls are managed by one Management Component, the TOE may also be considered to be distributed but the focus of the certification should be restricted to the simplest combination of Stateful Traffic Filter Firewall and Management Component. By the use of an equivalency argument, the combination of multiple Stateful

Traffic Filter Firewalls together with one Management Component can then be regarded as certified solution[3].

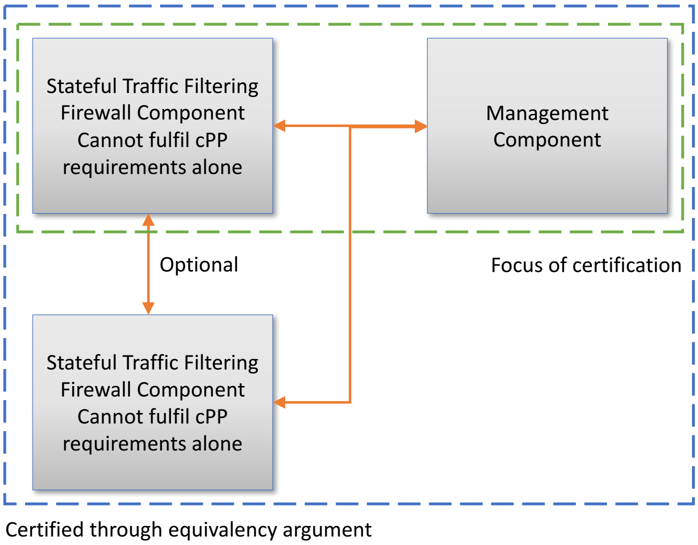


Figure 7: Distributed TOE extended through equivalency argument

In this model the individual Stateful Traffic Filter Firewall components rely on functionality within the Management Component to fulfil the requirements of this cPP and therefore a direct relationship between Stateful Traffic Filter Firewall components themselves is optional.

More than one Management Component may be used if it is for the sole purpose of redundancy.

3.2 Unsupported Distributed TOE Use Cases

The following discussion provides guidance for the distributed TOE use cases that are not supported by this version of the cPP.

[3] [SD-ND, B.4] describes how to define the components of a distributed TOE in terms of a “minimum configuration” and allowance for iteration of equivalent components.

Case 4: cPP requirements depend on using Management Component shared with other components outside the distributed TOE

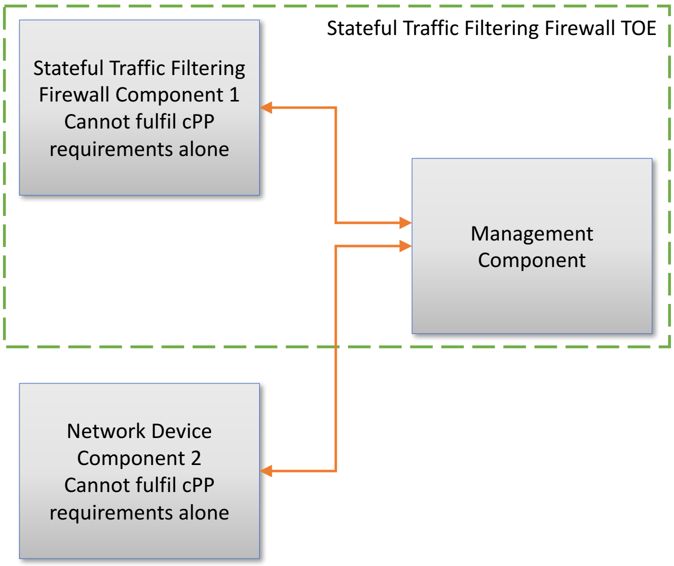


Figure 8: Unsupported Enterprise Management use case

Although apparently similar to Use Case 3 above, in this case a single Management Component is shared between the distributed Stateful Traffic Filter Firewall TOE and another distinct product (**Error! Reference source not found.** shows an example in which the other product is a network device). In this case the Management Component is considered to be an "Enterprise Manager" (a central management component for different types of devices), and this use case is not supported by this version of the cPP. A similar situation would apply if any other Stateful Traffic Filter Firewall TOE component was shared with another product.

Case 5: cPP requirements cannot be fulfilled without multiple Management Components

The case where one device, distributed TOE or combination of TOEs according to Case 3 above are managed by more than one Management Component (except for the purpose of redundancy) is not covered by this version of the cPP.

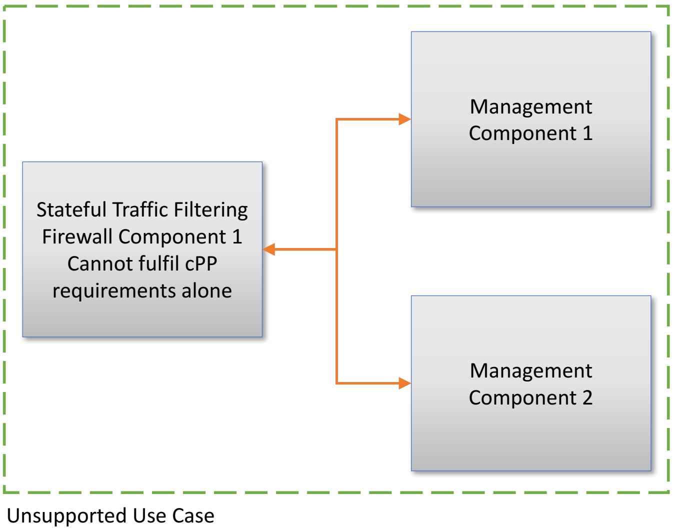


Figure 9: Unsupported use case with Multiple Management Components

3.3 Registration of components of a distributed TOE

When dealing with a distributed TOE, a number of separate components need to be brought together in the operational environment in order to create the TOE: this requires that trusted communications channels are set up between certain pairs of components (it is assumed that all components need to communicate with at least one other component, but not that all components need to communicate with all other components).

The underlying model for creation of the TOE is to have a "registration process" in which components "join" the TOE. The registration process starts with two components, one of which (the "joiner") is about to join an existing TOE by registering with the other (the "gatekeeper"). The two components will use one or more specified authentication and communication channel options so that the components authenticate each other and protect any sensitive data that is transmitted during the registration process (e.g. a key might be sent by a gatekeeper to the joiner as a result of the registration). The following figures illustrate the three supported registration models. Figure 10 illustrates a distributed TOE registration approach which uses an instance of FPT_ITT.1 or FTP_ITC.1 to protect the registration exchange.

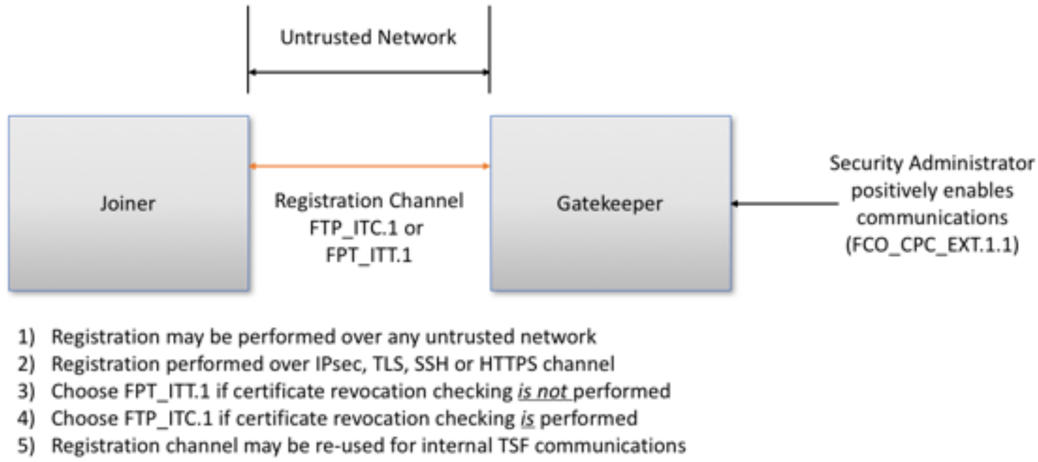


Figure 10 Distributed TOE registration using channel satisfying FPT_ITT.1 or FTP_ITC.1

The second approach (Figure 11) utilises an alternative registration channel and supports use-cases where the channel relies on environmental security constraints to provide the necessary protection of the registration exchange.

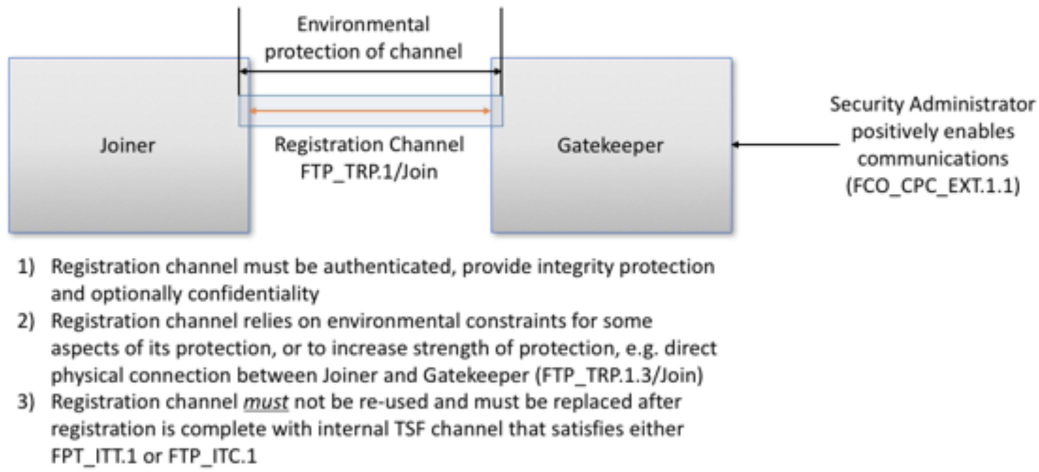


Figure 11 Distributed TOE registration using channel satisfying FTP_TRP.1/Join

The final approach (Figure 12) supports use-cases where registration is performed manually through direct configuration of both the Joiner and Gatekeeper devices. Once configured, the two components establish an internal TSF channel that satisfies FPT_ITT.1 or FTP_ITC.1.

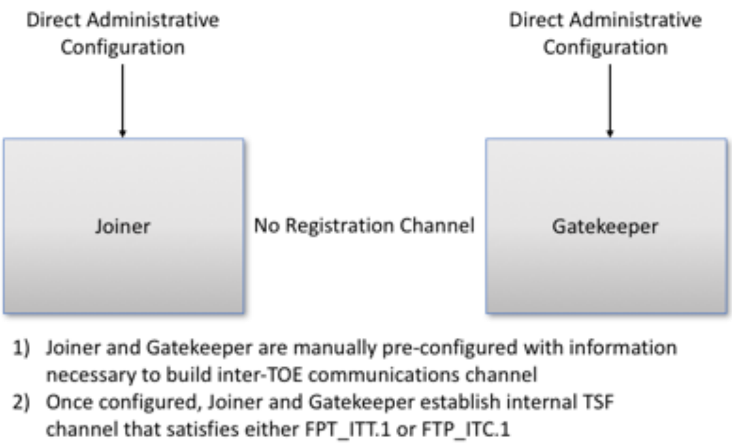


Figure 12 Distributed TOE registration without a registration channel

In each case, during the registration process, the Security Administrator must positively enable the joining components before it can act as part of the TSF. The following figure illustrates the approaches that this enablement step may take;

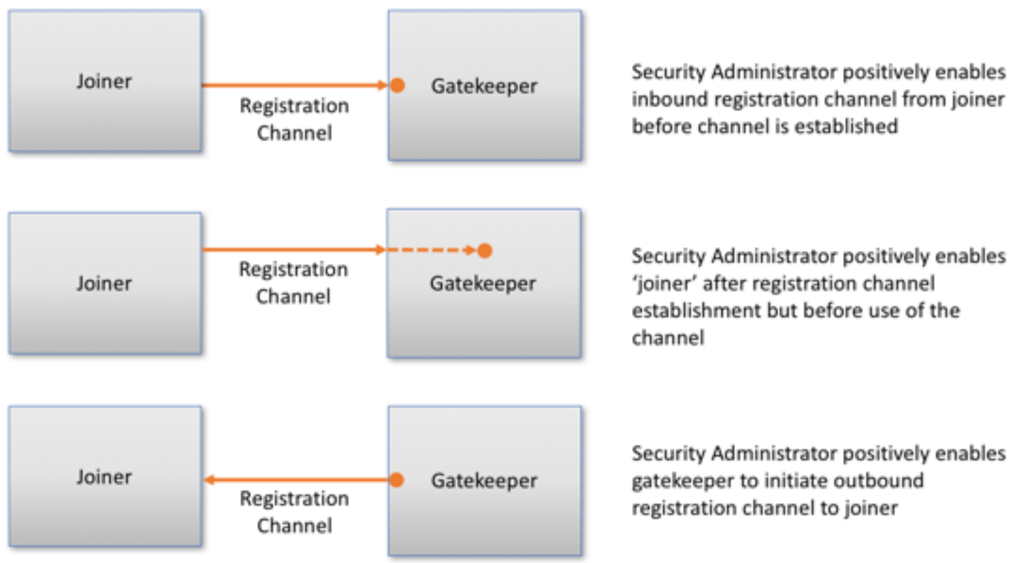


Figure 13 Joiner enablement options for Distributed TOEs

Note that in the case where no registration channel is required, that is the joiner and gatekeeper are directly configured (Figure 12), enablement is implied as part of this direct configuration process.

After registration the components will communicate between themselves using a normal SSH/TLS/DTLS/IPsec/HTTPS channel (which is specified in an ST as an instance of FTP_ITC.1 or FPT_ITT.1 in terms of section 6 and appendix A). This channel for inter-component communications is specified at the top level with the new (extended) SFR FCO_CPC_EXT.1 (see section A.7.1) and is in addition to the other communication channels required for communication with entities outside the TOE (which are specified in an ST as instances of FTP_ITC.1 and FTP_TRP.1).

3.4 Allocation of Requirements in Distributed TOEs

For a distributed TOE, the security functional requirements in this cPP need to be met by the TOE as a whole, but not all SFRs will necessarily be implemented by all components. The following categories are defined in order to specify when each SFR must be implemented by a component:

- **All Components (“All”)** – All components that comprise the distributed TOE must independently satisfy the requirement.
- **At least one Component (“One”)** – This requirement must be fulfilled by at least one component within the distributed TOE.
- **Feature Dependent (“Feature Dependent”)** – These requirements will only be fulfilled where the feature is implemented by the distributed TOE component (note that the requirement to meet the cPP as a whole requires that at least one component implements these requirements if they are specified in section 6).

Table 1 specifies how each of the SFRs in this cPP must be met, using the categories above.

Requirement	Description	Distributed TOE SFR Allocation
FAU_GEN.1	Audit Data Generation	All
FAU_GEN.2	User Identity Association	All
FAU_STG_EXT.1	Protected Audit Event Storage	All
FAU_STG.1	Protected Audit Trail Storage	Feature Dependent
FAU_STG_EXT.2/LocSpace	Counting Lost Audit Data	Feature Dependent
FAU_STG.3/LocSpace	Display warning for local storage space	Feature Dependent
FCO_CPC_EXT.1	Communication Partner Control	All

Requirement	Description	Distributed TOE SFR Allocation
FCS_CKM.1	Cryptographic Key Generation	One[4]
FCS_CKM.2	Cryptographic Key Establishment	All
FCS_CKM.4	Cryptographic Key Destruction	All
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)	All
FCS_COP.1/SigGen	Cryptographic Operation (Signature Verification)	All
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)	All
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)	All
FCS_DTLSC_EXT.1	DTLS client	Feature Dependent
FCS_DTLSC_EXT.2	DTLS client with mutual authentication	Feature Dependent
FCS_DTLSS_EXT.1	DTLS server	Feature Dependent
FCS_DTLSS_EXT.2	DTLS server with mutual authentication	Feature Dependent
FCS_HTTPS_EXT.1	HTTPS Protocol	Feature Dependent
FCS_IPSEC_EXT.1	IPsec Protocol	Feature Dependent

[4] Each component of a distributed TOE will be required either to perform on-board key generation and (if the TOE uses X.509 certificates as in Appendix B.3.1) RFC 2986 Certificate Request generation, or else to receive its keys and certificates, generated on some other component of the TOE, using a secure registration channel at the point where the component is joined to the TOE. (subsequent changes of keys and certificates may then use the post-registration inter-component secure channel). Certificate request generation will be required from either the component that generates the key or the component that receives the key.

Requirement	Description	Distributed TOE SFR Allocation
FCS_SSHC_EXT.1	SSH Client	Feature Dependent
FCS_SSHS_EXT.1	SSH Server	Feature Dependent
FCS_TLSC_EXT.1	TLS Client	Feature Dependent
FCS_TLSC_EXT.2	TLS Client with authentication	Feature Dependent
FCS_TLSS_EXT.1	TLS Server	Feature Dependent
FCS_TLSS_EXT.2	TLS Server with mutual authentication	Feature Dependent
FCS_RBG_EXT.1	Random Bit Generation	All
FDP_RIP.2	Full Residual Information Protection	Feature Dependent
FIA_AFL.1	Authentication Failure Management	One
FIA_PMG_EXT.1	Password Management	One
FIA_UIA_EXT.1	User Identification and Authentication	One
FIA_UAU_EXT.2	Password-based Authentication Mechanism	One
FIA_UAU.7	Protected Authentication Feedback	Feature Dependent
FIA_X509_EXT.1/Rev	X.509 Certification Validation	Feature Dependent
FIA_X509_EXT.1/ITT	X.509 Certification Validation	Feature Dependent
FIA_X509_EXT.2	X.509 Certificate Authentication	Feature Dependent
FIA_X509_EXT.3	Certificate Requests	Feature Dependent[4]

Requirement	Description	Distributed TOE SFR Allocation
FMT_MOF.1/ManualUpdate	Trusted Update - Management of Security Functions behaviour	All
FMT_MOF.1/Services	Trusted Update - Management of TSF Data	Feature Dependent
FMT_MOF.1/Functions	Management of security functions behaviour	Feature Dependent
FMT_MTD.1/CoreData	Management of TSF Data	All
FMT_MTD.1/CryptoKeys	Management of TSF Data	Feature Dependent
FMT_SMF.1	Specification of Management Functions	Feature Dependent
FMT_SMR.2	Restrictions on Security Roles	One
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all symmetric keys)	All
FPT_APW_EXT.1	Protection of Administrator Passwords	Feature Dependent
FPT_TST_EXT.1	Testing (Extended)	All
FPT_ITT.1	Basic internal TSF data transfer protection	Feature Dependent[5]
FPT_STM_EXT.1	Reliable Time Stamps	All
FPT_TST_EXT.2	Self-Test Based on Certificates	Feature Dependent

[5] To protect inter-TSF data transfer, FPT_ITT.1 or FTP_ITC.1 must be fulfilled by each distributed TOE component. This is in addition to an iteration of FTP_ITC.1 to protect communications with external entities.

Requirement	Description	Distributed TOE SFR Allocation
FPT_TUD_EXT.1	Trusted Update	All
FPT_TUD_EXT.2	Trusted Update based on Certificates	Feature Dependent
FTA_SSL.3	TSF-initiated Termination	Feature Dependent
FTA_SSL.4	User-Initiated Termination	Feature Dependent
FTA_SSL_EXT.1	TSF-Initiated Session Locking	Feature Dependent
FTA_TAB.1	Default TOE Access Banner	One
FTP_ITC.1	Inter-TSF Trusted Channel (Refinement)	One
FTP_TRP.1/Admin	Trusted Path (Refinement)	One
FTP_TRP.1/Join	Trusted Path	Feature Dependent
FMT_MOF.1/ManualUpdate	Management of security functions behaviour	Feature Dependent
FMT_MOF.1/AutoUpdate	Management of security functions behaviour	Feature Dependent
FFW_RUL_EXT.1	Stateful traffic filtering	One
FFW_RUL_EXT.2	Stateful filtering of dynamic protocols	Feature Dependent

Table 1: Security Functional Requirements for Distributed TOEs

The ST for a distributed TOE must include a mapping of SFRs to each of the components of the TOE. (Note that this deliverable is examined as part of the ASE_TSS.1 and AVA_VAN.1 Evaluation Activities as described in [SD-ND, 5.1.2] and [SD-ND, 5.6.1.1] respectively.) The ST for a distributed TOE may also introduce a "minimum configuration" and identify components that may have instances added to an operational configuration without affecting the validity of the CC certification. [SD-ND, B.4] describes Evaluation Activities relating to these equivalency aspects of a distributed TOE (and hence what is expected in the ST).

4. Security Problem Definition

A Stateful Traffic Filter Firewall (defined to be a device that filters layers 3 and 4 (IP and TCP/UDP) network traffic optimized through the use of stateful packet inspection) is intended to provide a minimal, baseline set of requirements that are targeted at mitigating well defined and described threats.

It has the ability to match packets to a known active (and allowed) connection to permit them and drop others. The firewall often serves as a boundary device between two separate network security domains, and, as such, must provide a minimal set of common security functionality. These functional requirements define authorized communication with the firewall, audit capabilities, user access, update processes, and self-test procedures for critical components.

4.1 Threats

The threats for the Stateful Traffic Filter Firewall are grouped according to functional areas of the device in the sections below. The description of each threat is then followed by a rationale describing how it is addressed by the SFRs in section 6, appendix A, and appendix B.

4.1.1 Communications with the Firewall

A firewall communicates with other network devices and other network entities. The endpoints of this communication can be geographically and logically distant and may pass through a variety of other systems. The intermediate systems may be untrusted providing an opportunity for unauthorized communication with the firewall or for authorized communication to be compromised. The security functionality of the firewall must be able to protect any critical network traffic (administration traffic, authentication traffic, audit traffic, etc.). The communication with the firewall falls into two categories: authorized communication and unauthorized communication.

Authorized communication includes normal network traffic allowable by policy destined to and originating from the firewall as it was designed and intended. This includes critical network traffic, such as firewall administration and communication with an authentication or audit logging server, which requires a secure channel to protect the communication. The security functionality of the firewall includes the capability to ensure that only authorized communications are allowed and the capability to provide a secure channel for critical network traffic. Any other communication is considered unauthorized communication.

The primary threats to firewall communications addressed in this cPP focus on an external, unauthorized entity attempting to access, modify, or otherwise disclose the critical network traffic. A poor choice of cryptographic algorithms or the use of non-standardized tunnelling protocols along with weak administrator credentials, such as an easily guessable password or use of a default password, will allow a threat agent unauthorized access to the firewall. Weak or no cryptography provides little to no protection of the traffic allowing a threat agent to read, manipulate and/or control the critical data with little effort. Non-standardized tunnelling protocols not only limit the interoperability of the firewall but lack the assurance and confidence standardization provides through peer review.

4.1.1.1 T.UNAUTHORIZED_ADMINISTRATOR_ACCESS

Threat agents may attempt to gain administrator access to the firewall by nefarious means such as masquerading as an administrator to the firewall, masquerading as the firewall to an administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between the firewall and a network device. Successfully gaining administrator access allows malicious actions that compromise the security functionality of the firewall and the network on which it resides.

SFR Rationale:

- The Administrator role is defined in FMT_SMR.2 and the relevant administration capabilities are defined in FMT_SMF.1 and FMT_MTD.1/CoreData, with optional additional capabilities in FMT_MOF.1/Services and FMT_MOF.1/Functions
- The actions allowed before authentication of an Administrator are constrained by FIA_UIA_EXT.1, and include the advisory notice and consent warning message displayed according to FTA_TAB.1
- The requirement for the Administrator authentication process is described in FIA_UAU_EXT.2
- Locking of Administrator sessions is ensured by FTA_SSL_EXT.1 (for local sessions), FTA_SSL.3 (for remote sessions), and FTA_SSL.4 (for all interactive sessions)
- The secure channel used for remote Administrator connections is specified in FTP_TRP.1/Admin
- (Malicious actions carried out from an Administrator session are separately addressed by T.UNDETECTED_ACTIVITY)
- (Protection of the Administrator credentials is separately addressed by T.PASSWORD_CRACKING).

4.1.1.2 T.WEAK_CRYPTOGRAPHY

Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.

SFR Rationale:

- Requirements for key generation and key distribution are set in FCS_CKM.1 and FCS_CKM.2 respectively
- Requirements for use of cryptographic schemes are set in FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, and FCS_COP.1/KeyedHash
- Requirements for random bit generation to support key generation and secure protocols (see SFRs resulting from T.UNTRUSTED_COMMUNICATION_CHANNELS) are set in FCS_RBG_EXT.1

- Management of cryptographic functions is specified in FMT_SMF.1

4.1.1.3 T.UNTRUSTED_COMMUNICATION_CHANNELS

Threat agents may attempt to target firewalls that do not use standardized secure tunnelling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the firewall itself.

SFR Rationale:

- The general use of secure protocols for identified communication channels is described at the top level in FTP_ITC.1 and FTP_TRP.1/Admin; for distributed TOEs the requirements for inter-component communications are addressed by the requirements in FPT_ITT.1
- Requirements for the use of secure communication protocols are set for all the allowed protocols in FCS_DTLSC_EXT.1, FCS_DTLSC_EXT.2, FCS_DTLSS_EXT.1, FCS_DTLSS_EXT.2, FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.1, FCS_TLSS_EXT.2
- Optional and selection-based requirements for use of public key certificates to support secure protocols are defined in FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3

4.1.1.4 T.WEAK_AUTHENTICATION_ENDPOINTS

Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints – e.g. a shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the Administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the firewall itself could be compromised.

SFR Rationale:

- The use of appropriate secure protocols to provide authentication of endpoints (as in the SFRs addressing T.UNTRUSTED_COMMUNICATION_CHANNELS) are ensured by the requirements in FTP_ITC.1 and FTP_TRP.1/Admin; for distributed TOEs the authentication requirements for endpoints in inter-component communications are addressed by the requirements in FPT_ITT.1
- Additional possible special cases of secure authentication during registration of distributed TOE components are addressed by FCO_CPC_EXT.1 and FTP_TRP.1/Join.

4.1.2 Valid Updates

Updating firewall software and firmware is necessary to ensure that the security functionality of the firewall is maintained. The source and content of an update to be applied must be validated by cryptographic means; otherwise, an invalid source can write their own firmware or software updates that circumvents the security functionality of the firewall. Methods of validating the source and content of a software or firmware update by cryptographic means typically involve cryptographic signature schemes where hashes of the updates are digitally signed.

Unpatched versions of software or firmware leave the firewall susceptible to threat agents attempting to circumvent the security functionality using known vulnerabilities. Non-validated updates or updates validated using non-secure or weak cryptography leave the updated software or firmware vulnerable to threat agents attempting to modify the software or firmware to their advantage.

4.1.2.1 T.UPDATE_COMPROMISE

Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.

SFR Rationale:

- Requirements for protection of updates are set in FPT_TUD_EXT.1
- Additional optional use of certificate-based protection of signatures can be specified using FPT_TUD_EXT.2, supported by the X.509 certificate processing requirements in FIA_X509_EXT.1, FIA_X509_EXT.2 and FIA_X509_EXT.3
- Requirements for management of updates are defined in FMT_SMF.1 and (for manual updates) in FMT_MOF.1/ManualUpdate, with optional requirements for automatic updates in FMT_MOF.1/AutoUpdate

4.1.3 Audited Activity

Auditing of firewall activities is a valuable tool for Administrators to monitor the status of the device. It provides the means for Administrator accountability, security functionality activity reporting, reconstruction of events, and problem analysis. Processing performed in response to device activities may give indications of a failure or compromise of the security functionality. When indications of activity that impact the security functionality are not generated and monitored, it is possible for such activities to occur without Administrator awareness. Further, if records are not generated and retained, reconstruction of the network and the ability to understand the extent of any compromise could be negatively affected. Additional concerns are the protection of the audit data that is recorded from alteration or unauthorized deletion. This could occur within the TOE, or while the audit data is in transit to an external storage device.

Note this cPP requires that the firewall generate the audit data and have the capability to send the audit data to a trusted network entity (e.g., a syslog server).

4.1.3.1 T.UNDETECTED_ACTIVITY

Threat agents may attempt to access, change, and/or modify the security functionality of the firewall without Administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the Administrator would have no knowledge that the device has been compromised.

SFR Rationale:

- Requirements for basic auditing capabilities are specified in FAU_GEN.1 and FAU_GEN.2, with timestamps provided according to FPT_STM_EXT.1
- Requirements for protecting audit records stored on the TOE are specified in FAU_STG.1
- Requirements for secure transmission of local audit records to an external IT entity via a secure channel are specified in FAU_STG_EXT.1
- Optional additional requirements for dealing with potential loss of locally stored audit records are specified in FAU_STG_EXT.2/LocSpace, and FAU_STG.3/LocSpace
- If (optionally) configuration of the audit functionality is provided by the TOE then this is specified in FMT_SMF.1, and confining this functionality to Security Administrators is required by FMT_MOF.1/Functions.

4.1.4 Administrator and Firewall Credentials and Data

A firewall contains data and credentials which must be securely stored and must appropriately restrict access to authorized entities. Examples include the firewall firmware, software, configuration authentication credentials for secure channels, and Administrator credentials. Firewall and Administrator keys, key material, and authentication credentials need to be protected from unauthorized disclosure and modification. Furthermore, the security functionality of the device needs to require default authentication credentials, such as Administrator passwords, be changed.

Lack of secure storage and improper handling of credentials and data, such as unencrypted credentials inside configuration files or access to secure channel session keys, can allow an attacker to not only gain access to the firewall, but also compromise the security of the network through seemingly authorized modifications to configuration or though man-in-the-middle attacks. These attacks allow an unauthorized entity to gain access and perform administrative functions using the Security Administrator's credentials and to intercept all traffic as an authorized endpoint. This results in difficulty in detection of security compromise and in reconstruction of the network, potentially allowing continued unauthorized access to Administrator and firewall data.

4.1.4.1 T.SECURITY_FUNCTIONALITY_COMPROMISE

Threat agents may compromise credentials and firewall data enabling continued access to the firewall and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or firewall credentials for use by the attacker.

SFR Rationale:

- Protection of secret/private keys against compromise is specified in FPT_SKP_EXT.1
- Secure destruction of keys is specified in FCS_CKM.4
- If (optionally) management of keys is provided by the TOE then this is specified in FMT_SMF.1, and confining this functionality to Security Administrators is required by FMT_MTD.1/CryptoKeys
- (Protection of passwords is separately covered under T.PASSWORD_CRACKING),

4.1.4.2 T.PASSWORD_CRACKING

Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the firewall. Having privileged access to the firewall provides the attacker unfettered access to the network traffic, and may allow them to take advantage of any trust relationships with other network devices.

SFR Rationale:

- Requirements for password lengths and available characters are set in FIA_PMG_EXT.1
- Protection of password entry by providing only obscured feedback is specified in FIA_UAU.7
- Actions on reaching a threshold number of consecutive password failures are specified in FIA_AFL.1
- Requirements for secure storage of passwords are set in FPT_APW_EXT.1.

4.1.5 Firewall Component Failure

Security mechanisms of the firewall generally build up from roots of trust to more complex sets of mechanisms. Failures could result in a compromise to the security functionality of the firewall. A firewall self-testing its security critical components at both start-up and during run-time ensures the reliability of the firewall's security functionality.

4.1.5.1 T.SECURITY_FUNCTIONALITY_FAILURE

A component of the firewall may fail during start-up or during operations causing a compromise or failure in the security functionality of the firewall, leaving it susceptible to attackers.

SFR Rationale:

- Requirements for running self-test(s) are defined in FPT_TST_EXT.1
- Optional use of certificates to support self-test(s) is defined in FPT_TST_EXT.2 (with support for the use of certificates in FIA_X509_EXT.1, FIA_X509_EXT.2, and FIA_X509_EXT.3),

4.1.6 Unauthorized Disclosure of Information

Devices on a protected network may be exposed to threats presented by devices located outside the protected network, which may attempt to conduct unauthorized activities. If known malicious external devices are able to communicate with devices on the protected network, or if devices on the protected network can establish communications with those external devices, then those internal devices may be susceptible to the unauthorized disclosure of information.

From an infiltration perspective, Stateful Traffic Filter Firewalls serve to limit access to only specific *destination* network addresses and ports within a protected network. With these limits, general network port scanning can be prevented from reaching protected networks or machines, and access to information on a protected network can be limited to that obtainable from specifically configured ports on identified network nodes (e.g., web pages from a designated corporate web server). Additionally, access can be limited to only specific *source* addresses and ports so that specific networks or network nodes can be blocked from accessing a protected network thereby further limiting the potential disclosure of information.

From an exfiltration perspective, Stateful Traffic Filter Firewalls serve to limit how network nodes operating on a protected network can connect to and communicate with other networks limiting how and where they can disseminate information. Specific external networks can be blocked altogether or egress could be limited to specific addresses and/or ports. Alternately, egress options available to network nodes on a protected network can be carefully managed in order to, for example, ensure that outgoing connections are routed through authorized proxies or filters to further mitigate inappropriate disclosure of data through extrusion.

4.1.6.1 T.NETWORK_DISCLOSURE

An attacker may attempt to “map” a subnet to determine the machines that reside on the network, and obtaining the IP addresses of machines, as well as the services (ports) those machines are offering. This information could be used to mount attacks to those machines via the services that are exported.

SFR Rationale:

- Requirements to prevent unauthorised disclosure of network information are defined in FFW_RUL_EXT.1 and FFW_RUL_EXT.2

4.1.7 Inappropriate Access to Services

Devices located outside the protected network may seek to exercise services located on the protected network that are intended to only be accessed from inside the protected network. Devices located outside the protected network may, likewise, offer services that are inappropriate for access from within the protected network.

From an ingress perspective, Stateful Traffic Filter Firewalls can be configured so that only those network servers intended for external consumption are accessible and only via the intended ports. This serves to mitigate the potential for network entities outside a protected network to access network servers or services intended only for consumption or access inside a protected network.

From an egress perspective, Stateful Traffic Filter Firewalls can be configured so that only specific external services (e.g., based on destination port) can be accessed from within a protected network. For example, access to external mail services can be blocked to enforce corporate policies against accessing uncontrolled e-mail servers. Note that the effectiveness of a Stateful Traffic Filter Firewall is rather limited in this regard since external servers can offer their services on alternate ports – this is where an Application Filter Firewall offers more reliable protection, for example.

4.1.7.1 T. NETWORK_ACCESS

With knowledge of the services that are exported by machines on a subnet, an attacker may attempt to exploit those services by mounting attacks against those services.

SFR Rationale:

- Requirements to prevent unauthorised access to protected devices and services are defined in FFW_RUL_EXT.1 and FFW_RUL_EXT.2

4.1.8 Misuse of Services

Devices located outside a "protected" network, while permitted to access particular *public* services offered inside the protected network, may attempt to conduct inappropriate activities while communicating with those allowed public services. Certain services offered from within a protected network may also represent a risk when accessed from outside the protected network. It should be noted that the firewall simply enforces rules that are specified for a network interface. The notion of a protected or trusted network is an abstraction that is useful when constructing the ruleset.

From an ingress perspective, it is generally assumed that entities operating on external networks are not bound by the use policies for a given protected network. Nonetheless, Stateful Traffic Filter Firewalls can log policy violations that might indicate violation of publicized usage statements for publicly available services.

From an egress perspective, Stateful Traffic Filter Firewalls can be configured to help enforce and monitor protected network use policies. As explained in the other threats, a Stateful Traffic Filter Firewall can serve to limit dissemination of data, access to external servers, and even disruption of services – all of these could be related to the use policies of a protected network and as such are subject in some regards to enforcement. Additionally, Stateful Traffic Filter Firewalls can be configured to log network usages that cross between protected and external networks and as a result can serve to identify potential usage policy violations.

4.1.8.1 T.NETWORK_MISUSE

An attacker may attempt to use services that are exported by machines in a way that is unintended by a site's security policies. For example, an attacker might be able to use a service to "anonymize" the attacker's machine as they mount attacks against others.

SFR Rationale:

- Requirements to prevent network misuse traffic are defined in FFW_RUL_EXT.1 and FFW_RUL_EXT.2
- Requirements to prevent the unintended dissemination of data from packets after deletion are defined in FDP_RIP.2

4.1.9 Malicious Traffic

A Stateful Traffic Filter Firewall also provides protections against malicious or malformed packets. It will protect against attacks like modification of connection state information and replay attacks. These attacks could cause the firewall, or the devices it protects, to grant unauthorized access or even create a Denial of Service.

4.1.9.1 T.MALICIOUS_TRAFFIC

An attacker may attempt to send malformed packets to a machine in hopes of causing the network stack or services listening on UDP/TCP ports of the target machine to crash.

SFR Rationale:

- Requirements to prevent malformed traffic are defined in FFW_RUL_EXT.1

4.2 Assumptions

This section describes the assumptions made in identification of the threats and security requirements for firewall devices. The firewall is not expected to provide assurance in any of these areas, and as a result, requirements are not included to mitigate the threats associated.

4.2.1 A.PHYSICAL_PROTECTION

The firewall device 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 firewall's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the firewall and the data it contains. As a result, the cPP will not include any requirements on physical tamper protection or other physical attack mitigations. The cPP will not expect the product to defend against physical access to the firewall that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the firewall.

[OE.PHYSICAL]

4.2.2 A.LIMITED_FUNCTIONALITY

The firewall device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the firewall device should not provide a computing platform for general purpose applications (unrelated to networking/filtering functionality).

[OE.NO_GENERAL_PURPOSE]

4.2.3 A.TRUSTED_ADMINISTRATOR

The Security Administrator(s) for the firewall device are assumed to be trusted and to act in the best interest of security for the organization. This includes being appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the firewall. The firewall device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device.

[OE.TRUSTED_ADMIN]

4.2.4 A.REGULAR_UPDATES

The firewall device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

[OE.UPDATES]

4.2.5 A.ADMIN_CREDENTIALS_SECURE

The Administrator's credentials (private key) used to access the firewall device are protected by the platform on which they reside.

[OE.ADMIN_CREDENTIALS_SECURE]

4.2.6 A.COMPONENTS_RUNNING (applies to distributed TOEs only)

For distributed TOEs it is assumed that the availability of all TOE components is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. It is also assumed that in addition to the availability of all components it is also checked as appropriate that the audit functionality is running properly on all TOE components.

[OE.COMPONENTS_RUNNING]

4.2.7 A.RESIDUAL_INFORMATION

The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on firewall equipment when the equipment is discarded or removed from its operational environment.

[OE.RESIDUAL_INFORMATION]

4.3 Organizational Security Policy

An organizational security policy is a set of rules, practices, and procedures imposed by an organization to address its security needs. The description of each policy is then followed by a rationale describing how it is addressed by the SFRs in section 6, appendix A, and appendix B.

4.3.1 P.ACCESS_BANNER

The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.

SFR Rationale:

- An advisory notice and consent warning message is required to be displayed by FTA_TAB.1

5. Security Objectives

5.1 Security Objectives for the Operational Environment

The following subsections describe objectives for the Operational Environment.

5.1.1 OE.PHYSICAL

Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

5.1.2 OE.NO_GENERAL_PURPOSE

There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.

5.1.3 OE.TRUSTED_ADMIN

Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner.

5.1.4 OE.UPDATES

The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

5.1.5 OE.ADMIN_CREDENTIALS_SECURE

The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.

5.1.6 OE.COMPONENTS_RUNNING (applies to distributed TOEs only)

For distributed TOEs the Security Administrator ensures that the availability of every TOE component is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. The Security Administrator also ensures that it is checked as appropriate for every TOE component that the audit functionality is running properly.

5.1.7 OE.RESIDUAL_INFORMATION

The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

6. Security Functional Requirements

The individual security functional requirements are specified in the sections below. SFRs in this section are mandatory SFRs that any conformant TOE must meet. 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 a distributed TOE, the ST author should reference Table 1 for guidance on how each SFR should be met. The table details whether SFRs should be met by all TOE components, by at least one TOE component or whether they are dependent upon the feature being implemented by the TOE component. The ST for a distributed TOE must include a mapping of SFRs to each of the components of the TOE. (Note that this deliverable is examined as part of the ASE_TSS.1 and AVA_VAN.1 Evaluation Activities as described in [SD-ND, 5.1.2] and [SD-ND, 5.6.1.1] respectively.

The Evaluation Activities defined in [SD-ND] and [SD-FW] describe actions that the evaluator will take in order 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.

6.1 Conventions

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

- Unaltered SFRs are stated in the form used in [CC2] or their extended component definition (ECD);
- Refinement made in the PP: the refinement text is indicated with **bold text** and ~~strikethroughs~~;
- Selection wholly or partially completed in the PP: the selection values (i.e. the selection values adopted in the PP or the remaining selection values available for the ST) are indicated with underlined text

  e.g. "[selection: *disclosure, modification, loss of use*]" in [CC2] or an ECD might become "disclosure" (completion) or "[selection: disclosure, modification]" (partial completion) in the PP;
- Assignment wholly or partially completed in the PP: indicated with *italicized text*;
- Assignment completed within a selection in the PP: the completed assignment text is indicated with *italicized and underlined text*

  e.g. "[selection: *change_default, query, modify, delete, [assignment: other operations]*]" in [CC2] or an ECD might become "change_default, *select_tag*" (completion of both selection and assignment) or "[selection: change_default, *select_tag*, *select_value*]" (partial completion of selection, and completion of assignment) in the PP;
- Iteration: indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").

Extended SFRs are identified by having a label "EXT" at the end of the SFR name.

Where compliance to RFCs is referred to in SFRs, this is intended to be demonstrated by completing the corresponding Evaluation Activities in [SD-FW] and [SD-ND] for the relevant SFR.

6.2 SFR Architecture

Error! Reference source not found., Figure 15, Figure 16, Figure 17, Figure 18, Figure 19 and Figure 20 give a graphical presentation of the connections between the Security Functional Requirements in sections 6.3-10, Appendix A and Appendix B, and the underlying functional areas and operations that the TOE provides. The diagrams provide a context for SFRs that relates to their use in the TOE, whereas other sections define the SFRs grouped by the abstract class and family groupings in [CC2].

In the diagrams, the SFRs from Appendix B are both described as "Discretionary", meaning that their inclusion in an ST will depend on the particular properties of a product. The SFRs from Appendix B that are required by an ST are determined by the selections made in other SFRs. For example: FTP_ITC.1 and FTP_TRP.1/Admin (in sections 6.10.1.1 and 6.10.2.1 respectively) each contain selections of a protocol to be used for the type of secure channel described by the SFR. The selection of the protocol(s) here determines which of the protocol-specific SFRs in section B.2.1 are also required in the ST. SFRs in Appendix A can be included in the ST if they are provided by the TOE, but are not mandatory in order for a TOE to claim conformance to this cPP.

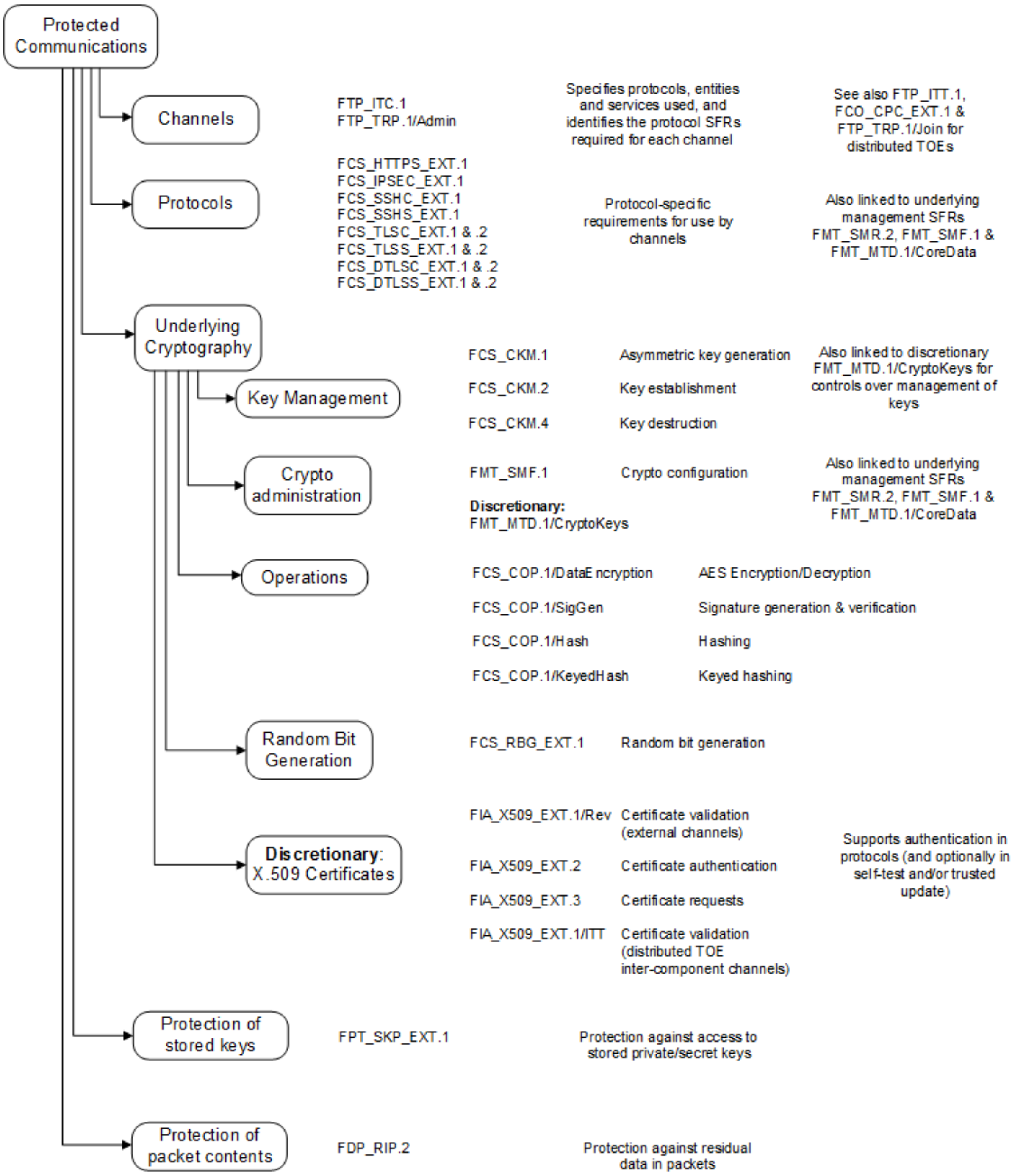


Figure 14: Protected Communications SFR Architecture

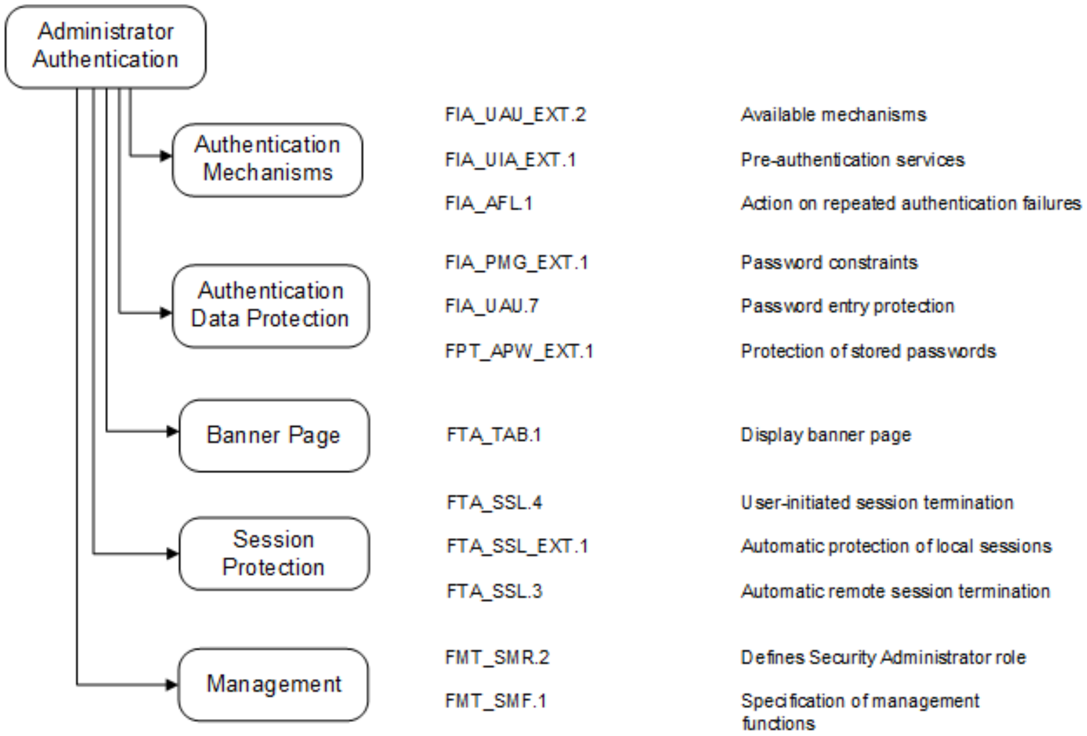


Figure 15: Administrator Authentication SFR Architecture

Figure 16: Correct Operation SFR Architecture

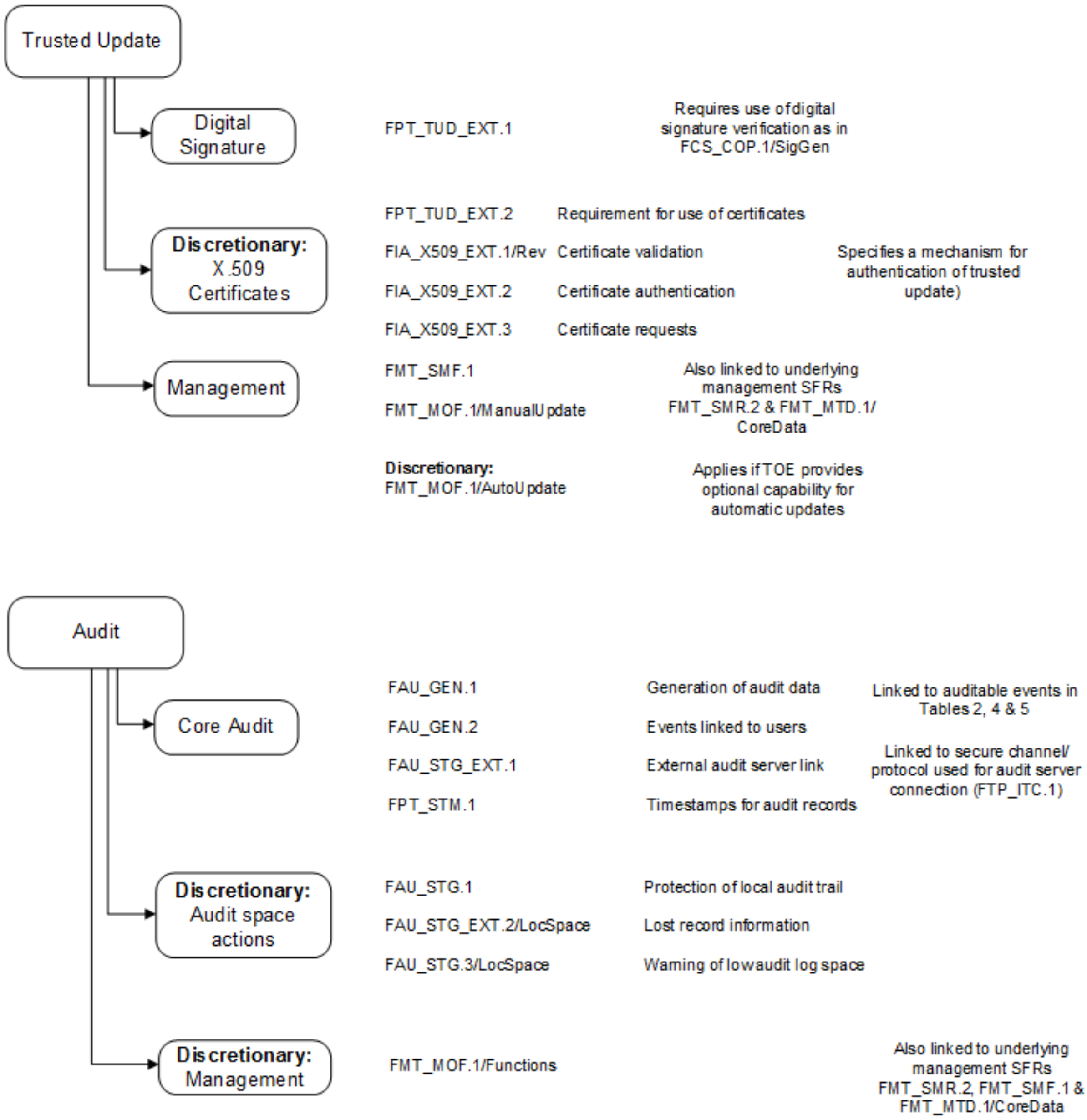


Figure 17: Trusted Update and Audit SFR Architecture

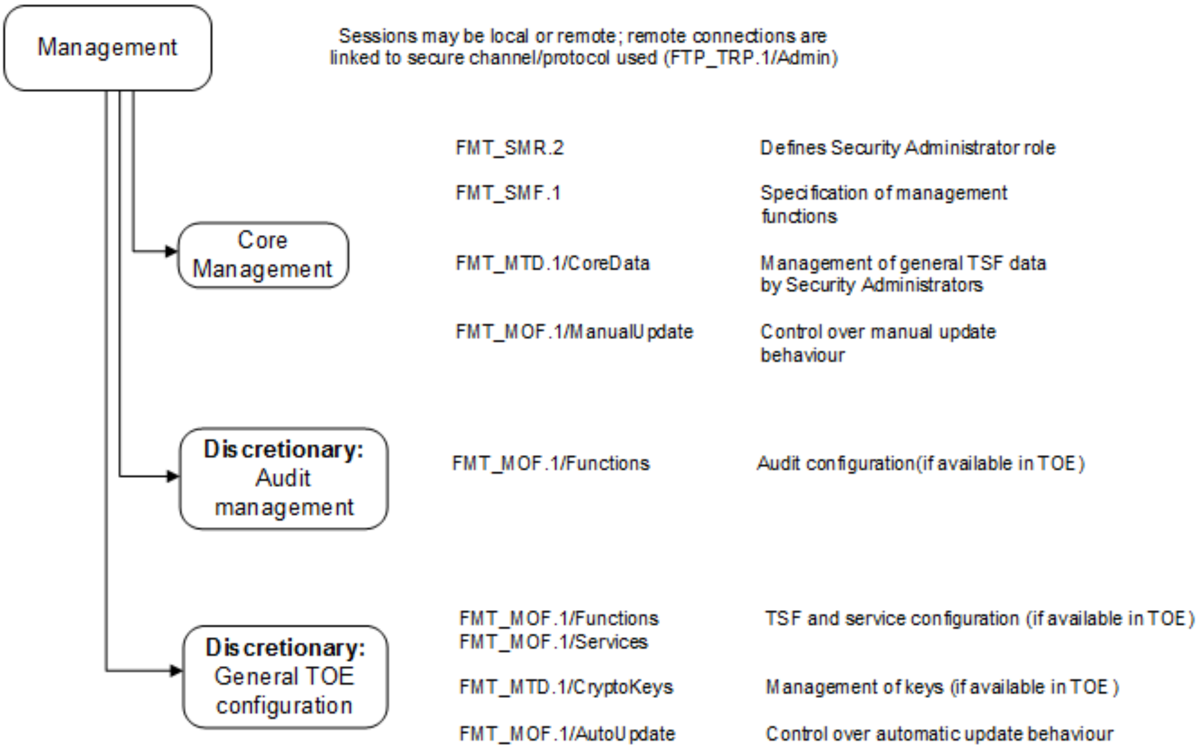


Figure 18: Management SFR Architecture

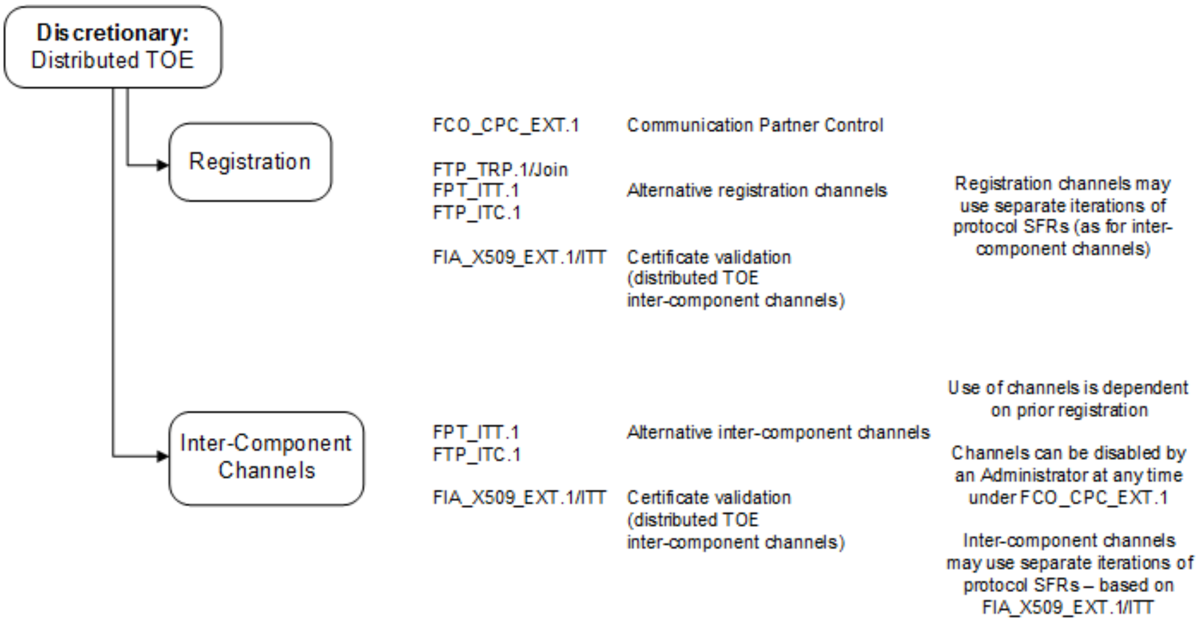


Figure 19: Distributed TOE SFR Architecture

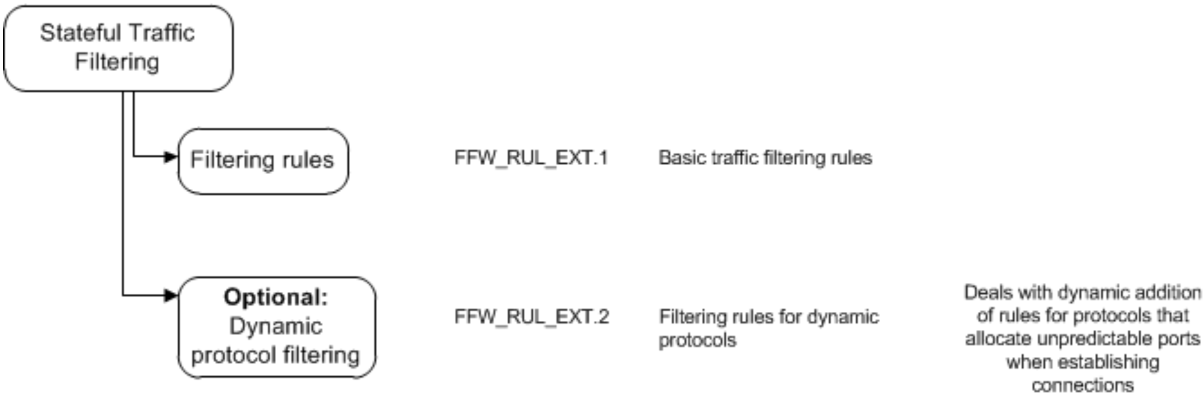


Figure 20: Firewall Rules SFR Architecture

6.3 Security Audit (FAU)

6.3.1 Security Audit Data generation (FAU_GEN)

In order to assure that information exists that allows Security Administrators to discover intentional and unintentional issues with the configuration and/or operation of the system, compliant TOEs have the capability of generating audit data targeted at detecting such activity. Auditing of administrative activities provides information that may be used to hasten corrective action should the system be configured incorrectly. Audit of select system events can provide an indication of failure of critical portions of the TOE (e.g. a cryptographic provider process not running) or anomalous activity (e.g. establishment of an administrative session at a suspicious time, repeated failures to establish sessions or authenticate to the system) of a suspicious nature.

In some instances, there may be a large amount of audit information produced that could overwhelm the TOE or Administrators in charge of reviewing the audit information. The TOE must be capable of sending audit information to an external trusted entity. This information must carry reliable timestamps, which will help order the information when sent to the external device.

Loss of communication with the audit server is problematic. While there are several potential mitigations to this threat, this cPP does not mandate that a specific action takes place; the degree to which this action preserves the audit information and still allows the TOE to meet its functionality responsibilities should drive decisions on the suitability of the TOE in a particular environment.

6.3.1.1 FAU_GEN.1 Audit data generation

FAU_GEN.1	Audit Data Generation

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

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

b) All auditable events for the not specified level of audit; and

c) *All administrative actions comprising:*

- *Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).*
- *Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).*
- *Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).*
- *Resetting passwords (name of related user account shall be logged).*
- *[selection: [Starting and stopping services, no other actions, assignment: [list of other uses of privileges]]];*

d) *Specifically defined auditable events listed in Table 2.*

Application Note 1

If the list of "administrative actions" appears to be incomplete, the assignment in the selection should be used to list additional administrative actions which are audited.

The ST author replaces the cross-reference to the table of audit events with an appropriate cross-reference for the ST. This must also include the relevant parts of Table 4 and Table 5 for optional and selection-based SFRs included in the ST.

For distributed TOEs each component must generate an audit record for each of the SFRs that it implements. If more than one TOE component is involved when an audit event is triggered, the event has to be audited on each component (e.g. rejection of a connection by one component while attempting to establish a secure communication channel between two components should result in an audit event being generated by both components). This is not limited to error cases but also includes events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

Application Note 2

The ST author can include other auditable events directly in the table; they are not limited to the list presented.

The TSS should identify what information is logged to identify the relevant key for the administrative task of generating/import of, changing, or deleting of cryptographic keys.

With respect to FAU_GEN.1.1 the term "services" refers to trusted path and trusted channel communications, on demand self-tests, trusted update and Administrator sessions (that exist under the trusted path) (e.g. netconf). If the optional SFR FMT_MOF.1/Services is included in the ST, the option "starting and stopping services" needs to be chosen from the selection in FAU_GEN.1.1.

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the cPP/ST, *information specified in column three of Table 2*.

Application Note 3

The ST author replaces the cross-reference to the table of audit events with an appropriate cross-reference for the ST. This must also include the relevant parts of Table 4 and Table 5 for optional and selection-based SFRs included in the ST.

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG_EXT.1	None.	None.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_RBG_EXT.1	None.	None.
FDP_RIP.2	None.	None.
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).

Requirement	Auditable Events	Additional Audit Record Contents
FIA_UAU.7	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	All management activities of TSF data.	None.
FMT_SMF.1	None.	None.
FMT_SMR.2	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1 (if “lock the session” is selected)	Any attempts at unlocking of an interactive session.	None.
FTA_SSL_EXT.1 (if “terminate the session” is selected)	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.
FFW_RUL_EXT.1	Application of rules configured with the 'log' operation	Source and destination addresses Source and destination ports Transport Layer Protocol TOE Interface
	Indication of packets dropped due to too much network traffic	TOE interface that is unable to process packets Identifier of rule causing packet drop

Table 2: Security Functional Requirements and Auditable Events

Application Note 4

Additional audit events will apply to the TOE depending on the optional and selection-based requirements adopted from Appendix A and Appendix B. The ST author must therefore include the relevant additional events specified in the tables in Table 4 and Table 5.

6.3.1.2 FAU_GEN.2 User identity association

FAU_GEN.2	User identity association

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

Application Note 5

Where an auditable event is triggered by another component, the component that records the event must associate the event with the identity of the initiating component that caused the event (applies to distributed TOEs only).

6.3.2 Security audit event storage (Extended – FAU_STG_EXT)

A network device TOE is not expected to take responsibility for all audit storage itself. Although it is required to store data locally at the time of generation, and to take some appropriate action if this local storage capacity is exceeded, the TOE is also required to be able to establish a secure link to an external audit server to enable external audit trail storage.

6.3.2.1 FAU_ STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1	Protected Audit Event Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1.

Application Note 6

For selecting the option of transmission of generated audit data to an external IT entity the TOE relies on a non-TOE audit server for storage and review of audit records. The storage of these audit records and the ability to allow the Administrator to review these audit records is provided by the operational environment in that case. Since the external audit server is not part of the TOE, there are no requirements on it except the capabilities for FTP_ITC.1 transport for audit data. No requirements are placed upon the format or underlying protocol of the audit data being transferred. The TOE must be capable of being configured to transfer audit data to an external IT entity without Administrator intervention. Manual transfer would not meet the requirements. Transmission could be done in real-time or periodically. If the transmission is not done in real-time then the TSS describes what event stimulates the transmission to be made and what range of frequencies the TOE supports for making transfers of audit data to the audit server; the TSS also suggests typical acceptable frequencies for the transfer.

For distributed TOEs each component must be able to export audit data across a protected channel external (FTP_ITC.1) or intercomponent (FPT_ITT.1 or FTP_ITC.1) as appropriate. At least one component of the TOE must be able to export audit records via FTP_ITC.1 such that all TOE audit records can be exported to an external IT entity.

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself.

FAU_STG_EXT.1.3 The TSF shall [selection: drop new audit data, overwrite previous audit records according to the following rule: *[assignment: rule for overwriting previous audit records]*, *[assignment: other action]*] when the local storage space for audit data is full.

Application Note 7

The external log server might be used as alternative storage space in case the local storage space is full. The "other action" could in this case be defined as "send the new audit data to an external IT entity".

For distributed TOEs each component is not required to store generated audit data locally but the overall TOE needs to be able to store audit data locally. Each component must at least provide the ability to temporarily buffer audit information locally to ensure that audit records are preserved in case of network connectivity issues (for details see also chap. 6.3.3). Buffering audit information locally, does not necessarily involve non-volatile memory: audit information could be buffered in volatile memory. However, the local storage of audit information in the sense of FAU_STG_EXT.1.3 needs to be done in non-volatile memory. For every component which performs local storage of audit information, the behaviour when local storage is exhausted needs to be described. For every component which is buffering audit information instead of storing audit information locally itself, it needs to be described what happens in case the buffer space is exhausted.

6.3.3 Security Audit for Distributed TOEs

For distributed TOEs the handling of audit information might be more complicated than for TOEs consisting only of one component. There are a few basic requirements to be fulfilled:

- Every component must be able to generate audit information.
- Every component must either be able to buffer audit information and forward it to another TOE component or to store audit information locally.
- For the overall TOE it must be possible to store all audit information locally.
- For the overall TOE it must be possible to send out audit information to an external audit server.

In general, every component must be able to generate its own audit information. It would be possible that every component also stores its own audit information locally as well as every component could be able to send out audit data to an external audit server. It would also be sufficient that every component would be able to generate its own audit data and buffer it locally before the information is sent out to one or more other TOE components for local storage and/or transmission to an external audit server. For the transfer of audit records between TOE components the secure connection via FTP_ITC.1 or FPT_ITT.1 must be used.

Such a solution would still be suitable to fulfil the requirement that all audit-related SFRs have to be fulfilled by all TOE components, although formally not every component would support local storage or transfer to an external audit server itself.

Regarding the establishment of inter-TOE communication, error conditions as well as successful connection/teardown events should be captured by both ends of the connection.

Although all TOE components shall be able to generate their own audit data according to FAU_GEN.1 for all the SFRs that they implement, not all TOE components have to provide audit data about all events. For distributed TOEs a mapping shall be provided to show which auditable events according to FAU_GEN.1 are covered by which components (also giving a justification that the records generated by each component cover all the SFRs that it implements). The overall TOE has to provide audit information about all events defined for FAU_GEN.1. As a result, at least one TOE component has to be assigned to every auditable event defined for FAU_GEN.1. The part of the mapping related to Table 2 shall be consistent with the mapping of SFRs to TOE components for ASE_TSS.1 in the sense that all components defined as generating audit information for a particular SFR should also contribute to that SFR in the mapping for ASE_TSS.1. This applies not only to audit events defined for mandatory SFRs but also to all audit events for optional and selection-based SFRs as defined in Appendix A and Appendix B.

If one or more of the optional audit components FAU_STG.1, FAU_STG_EXT.2/LocSpace and FAU_STG.3/LocSpace are selected in the Security Target derived from this cPP, then the SFR mapping for ASE_TSS.1 must include a specific identification of the TOE components to which they apply.

6.4 Cryptographic Support (FCS)

This section defines cryptographic requirements that underlie the other security properties of the TOE, covering key generation and random bit generation, key establishment methods, key destruction, and the various types of cryptographic operation to provide AES encryption/decryption, signature verification, hash generation, and keyed hash generation.

These SFRs support the implementation of the selection-based protocol-level SFRs in Appendix B.

6.4.1 Cryptographic Key Management (FCS_CKM)

6.4.1.1 FCS_CKM.1 Cryptographic Key Generation (Refinement)

FCS_CKM.1	**Cryptographic Key Generation**

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

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

- **FFC schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1**

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

Application Note 8

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 and selected cryptographic protocols must match the selection. When key generation is used for device authentication, other than ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384 and ecdsa-sha2-nistp521, the public key is expected to be associated with an X.509v3 certificate.

If the TOE acts as a receiver in the key establishment schemes and is not configured to support mutual authentication, the TOE does not need to implement key generation.

In a distributed TOE, if the TOE component acts as a receiver in the key establishment scheme, the TOE does not need to implement key generation.

6.4.1.2 FCS_CKM.2 Cryptographic Key Establishment (Refinement)

FCS_CKM.2	Cryptographic Key Establishment

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

- **RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B Revision 1, "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 Revision 2, "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 Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";**

- **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 9

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. For Diffie-Hellman group 14, ST authors should make the corresponding selection from the SFR instead of using the Finite field-based key establishment selection.

The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B ***Revision 1****; however, Section 9 relies on implementation of other sections in SP 800-56B* ***Revision 1****.*

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

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

6.4.1.3 FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4	Cryptographic Key Destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [*assignment:*

- *For plaintext keys in volatile storage, the destruction shall be executed by a [selection: single overwrite consisting of [selection: a pseudo-random pattern using the TSF's RBG, zeroes, ones, a new value of the key, [assignment: a static or dynamic value that does not contain any CSP]], destruction of reference to the key directly followed by a request for garbage collection];*

- *For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that [selection:*
  - *logically addresses the storage location of the key and performs a [selection: single, [assignment: number of passes]-pass] overwrite consisting of [selection: a pseudo-random pattern using the TSF's RBG, zeroes, ones, a new value of the key, [assignment: a static or dynamic value that does not contain any CSP]];*
  - *instructs a part of the TSF to destroy the abstraction that represents the key]]*

that meets the following: *No Standard*.

Application Note 10

In parts of the selections where keys are identified as being destroyed by "a part of the TSF", the TSS identifies the relevant part and the interface involved. The interface referenced in the requirement could take different forms for different TOEs, 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 another part of the TSF such as the interpreter or OS to delete the resource.

Where different key destruction methods are used for different keys and/or different destruction situations then the different methods and the keys/situations they apply to are described in the TSS (and the ST may use separate iterations of the SFR to aid clarity). The TSS describes all relevant keys used in the implementation of SFRs, including cases where the keys are stored in a non-plaintext form. In the case of non-plaintext storage, the encryption method and relevant key-encrypting-key are identified in the TSS.

Some selections allow assignment of "a value that does not contain any CSP". This means that the TOE uses some specified data not drawn from an RBG meeting FCS_RBG_EXT requirements, and not being any of the particular values listed as other selection options. The point of the phrase "does not contain any 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.

For the avoidance of doubt: the "cryptographic keys" in this SFR include session keys. Key destruction does not apply to the public component of asymmetric key pairs.

6.4.2 Cryptographic Operation (FCS_COP)

6.4.2.1 FCS_COP.1 Cryptographic Operation

FCS_COP.1/DataEncryption	**Cryptographic Operation (AES Data Encryption/ Decryption)**

FCS_COP.1.1/DataEncryption The TSF shall perform **encryption/decryption** in accordance with a specified cryptographic algorithm [*assignment: AES used in [selection: CBC, CTR, GCM] mode* and cryptographic key sizes *[assignment: 128 bits, 192 bits, 256 bits]* that meet the following: [*assignment: AES as specified in ISO 18033-3, [selection: CBC as specified in ISO 10116, CTR as specified in ISO 10116, GCM as specified in ISO 19772]*.

Application Note 11

For the first selection of FCS_COP.1.1/DataEncryption, the ST author chooses the mode or modes in which AES operates. For the second selection, the ST author chooses 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/SigGen	**Cryptographic Operation (Signature Generation and Verification)**

FCS_COP.1.1/SigGen 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 sizes **(modulus)** [*assignment: 2048 bits or greater*],
- Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [*assignment: 256 bits or greater]*

]

that meet the following: [selection:

- For RSA schemes: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,
- For ECDSA 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

].

Application Note 12

The ST Author chooses the algorithm(s) implemented to perform digital signatures. For the algorithm(s) chosen, the ST author makes the appropriate assignments/selections to specify the parameters that are implemented for that algorithm. The ST author ensures that the assignments and selections for this SFR include all the parameter values necessary for the cipher suites selected for the protocol SFRs (see Appendix B.2.1) that are included in the ST. The ST Author checks for consistency of selections with other FCS requirements, especially when supporting elliptic curves.

FCS_COP.1/Hash	**Cryptographic Operation (Hash Algorithm)**

FCS_COP.1.1/Hash The TSF shall perform **cryptographic hashing services** in accordance with a specified cryptographic algorithm [selection: SHA-1, SHA-256, SHA-384, SHA-512] ~~and cryptographic key sizes [*assignment: cryptographic key sizes*~~] and message digest sizes [selection: 160, 256, 384, 512] bits that meet the following: [*assignment: ISO/IEC 10118-3:2004]*.

Application Note 13

Vendors are strongly encouraged to implement updated protocols that support the SHA-2 family; until updated protocols are supported, this cPP allows support for SHA-1 implementations in compliance with SP 800-131A. In a future version of this cPP, SHA-256 will be the minimum requirement for all TOEs.

The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1/DataEncryption and FCS_COP.1/SigGen (for example, SHA 256 for 128-bit keys).

FCS_COP.1/KeyedHash	**Cryptographic Operation (Keyed Hash Algorithm)**

FCS_COP.1.1/KeyedHash The TSF shall perform **keyed-hash message authentication** in accordance with a specified cryptographic algorithm *[*selection: *HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512]* and cryptographic key sizes [*assignment: key size (in bits) used in HMAC*] **and message digest sizes [selection: 160, 256, 384, 512] bits** that meet the following: [*assignment: ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”]*.

Application Note 14

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.

6.4.3 Random Bit Generation (Extended – FCS_RBG_EXT)

6.4.3.1 FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1	Random Bit Generation

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [selection: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [selection: *[assignment: number of software-based sources]* software-based noise source, *[assignment: number of hardware-based sources]* hardware-based noise source] with a minimum of [selection: 128 bits, 192 bits, 256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011, of the keys and CSPs that it will generate.

Application Note 15

For the first selection in FCS_RBG_EXT.1.2, the ST author 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 Evaluation Activity for this element should be repeated to 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-1, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed.

If the key length for the AES implementation used here is different than that used to encrypt the user data, then FCS_COP.1 may have to be adjusted or iterated to reflect the different key length. For the selection in FCS_RBG_EXT.1.2, the ST author selects the minimum number of bits of entropy that is used to seed the RBG, which must be equal or greater than the security strength of any key generated by the TOE.

6.5 User Data Protection (FDP)

This section requires the TOE to ensure that it does not reuse old packet information when transmitting new packets.

6.5.1 Residual information protection (FDP_RIP)

6.5.1.1 FDP_RIP.2 Full Residual Information Protection

FDP_RIP.2	Full Residual Information Protection

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the [selection: *allocation of the resource to, deallocation of the resource from*] all objects.

Application Note 16

"Resources" in the context of this requirement are network packets being sent through (as opposed to "to", as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet.

6.6 Identification and Authentication (FIA)

In order to provide a trusted means for Administrators to interact with the TOE, the TOE provides a password-based logon mechanism. The Administrator must have the capability to compose a strong password, and have mechanisms in place so that the password must be changed regularly. To avoid attacks where an attacker might observe a password being typed by an Administrator, passwords must be obscured during logon. Session locking or termination must also be implemented to mitigate the risk of an account being used illegitimately. Passwords must be stored in an obscured form, and there must be no interface provided for specifically reading the password or password file such that the passwords are displayed in plain text.

6.6.1 Authentication Failure Management (FIA_AFL)

6.6.1.1 FIA_AFL.1 Authentication Failure Management (Refinement)

FIA_AFL.1	Authentication Failure Management

FIA_AFL.1.1 The TSF shall detect when **an Administrator configurable positive integer within** [*assignment: range of acceptable values*] unsuccessful authentication attempts occur related to **Administrators attempting to authenticate remotely**.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been **met**, the TSF shall [selection: prevent the offending remote Administrator from successfully authenticating until *[assignment: action]* is taken by a local Administrator*;* prevent the offending remote Administrator from successfully authenticating until an Administrator defined time period has elapsed*]*.

Application Note 17

This requirement applies to a defined number of successive unsuccessful authentication attempts and does not apply to an Administrator at the local console, since it does not make *sense to lock a local Administrator's account in this fashion. This could be addressed by (for example) requiring a separate account for local Administrators or having the authentication mechanism implementation distinguish local and remote login attempts. The "action" taken by a local Administrator is implementation specific and would be defined in the Administrator guidance (for example, lockout reset or password reset). The ST author chooses one of the selections for handling of authentication failures depending on how the TOE has implemented this handler.*

The TSS describes how the TOE ensures that authentication failures by remote Administrators cannot lead to a situation where no Administrator access is available, either permanently or temporarily (e.g. by providing local logon which is not subject to blocking). The Operational Guidance describes, and identifies the importance of, any actions that are required in order to ensure that Administrator access will always be maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

6.6.2 Password Management (Extended – FIA_PMG_EXT)

6.6.2.1 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1	Password Management

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [selection: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [*assignment: other characters*]];

b) Minimum password length shall be configurable to [*assignment: minimum number of characters supported by the TOE*] and [*assignment: number of characters greater than or equal to 15*].

Application Note 18

The ST author selects the special characters that are supported by the TOE. They may optionally list additional special characters supported using the assignment. "Administrative passwords" refers to passwords used by Administrators at the local console, over protocols that support passwords, such as SSH and HTTPS, or to grant configuration data that supports other SFRs in the Security Target.

The second assignment should be configured with the largest minimum password length the Security Administrator can configure.

6.6.3 User Identification and Authentication (Extended – FIA_UIA_EXT)

6.6.3.1 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1	User Identification and Authentication

FIA_UIA_EXT.1.1 The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- [selection: no other actions, *[assignment: list of services, actions performed by the TSF in response to non-TOE requests.]*]

FIA_UIA_EXT.1.2 The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

Application Note 19

This requirement applies to users (Administrators and external IT entities) of services available from the TOE directly, and not services available by connecting through the TOE. While it should be the case that few or no services are available to external entities prior to identification and authentication, if there are some available (perhaps ICMP echo) these should be listed in the assignment statement; otherwise "no other actions" should be selected.

Authentication can be password-based through the local console or through a protocol that supports passwords (such as SSH), or be certificate based (such as SSH, TLS).

For communications with external IT entities (an audit server, for instance), such connections must be performed in accordance with FTP_ITC.1, whose protocols perform identification and authentication. This means that such communications (e.g., establishing the IPsec connection to the authentication server) would not have to be specified in the assignment, since establishing the connection "counts" as initiating the identification and authentication process.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

6.6.4 User authentication (FIA_UAU) (Extended – FIA_UAU_EXT)

6.6.4.1 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2	Password-based Authentication Mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local password-based authentication mechanism, and [selection: *[assignment: other authentication mechanism(s)]*, no other authentication mechanism] to perform local administrative user authentication.

Application Note 20

The assignment should be used to identify any additional local authentication mechanisms supported. Local authentication mechanisms are defined as those that occur through the local console; remote administrative sessions (and their associated authentication mechanisms) are specified in FTP_TRP.1/Admin.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

6.6.4.2 FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7	**Protected Authentication Feedback**

FIA_UAU.7.1 The TSF shall provide only **obscured feedback** to the administrative user while the authentication is in progress at the local console.

Application Note 21

"Obscured feedback" implies the TSF does not produce a visible display of any authentication data entered by a user (such as the echoing of a password), although an obscured indication of progress may be provided (such as an asterisk for each character). It also implies that the TSF does not return any information during the authentication process to the user that may provide any indication of the authentication data.

6.7 Security Management (FMT)

Management functions required in this section describe required capabilities to support a Security Administrator role and basic set of security management functions dealing with management of configurable aspects included in other SFRs (FMT_SMF.1), general management of TSF data (FMT_MTD.1/CoreData), and enabling TOE updates (FMT_MOF.1/ManualUpdate).

For distributed TOEs security management of TOE components could be realized for every TOE component directly or through other TOE components. The TSS shall describe which management SFRs and management functions apply to each TOE component (applies only to distributed TOEs).

These core management requirements are supplemented by optional requirements in section A.4 and selection-based requirements in section B.5, according to the TOE capabilities.

6.7.1 Management of functions in TSF (FMT_MOF)

6.7.1.1 FMT_MOF.1/ManualUpdate Management of security functions behaviour

FMT_MOF.1/ManualUpdate	Management of security functions behaviour

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to **enable** the functions **to perform manual updates** to **Security Administrators**.

Application Note 22

FMT_MOF.1/ManualUpdate restricts the initiation of manual updates to Security Administrators.

6.7.2 Management of TSF Data (FMT_MTD)

6.7.2.1 FMT_MTD.1/CoreData Management of TSF Data

FMT_MTD.1/CoreData	Management of TSF Data

FMT_MTD.1.1/CoreData The TSF shall restrict the ability to **manage** the **TSF data** to **Security Administrators**.

Application Note 23

The word "manage" includes but is not limited to create, initialize, view, change default, modify, delete, clear, and append. This SFR includes also the resetting of user passwords by the Security Administrator. The identifier "CoreData" has been added here to separate this iteration of FMT_MTD.1 from the optional iteration of FMT_MTD.1 defined in Appendix A.4.2.1 (FMT_MTD.1/CryptoKeys).

6.7.3 Specification of Management Functions (FMT_SMF)

6.7.3.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1	Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

[*assignment:*

- *Ability to administer the TOE locally and remotely;*
- *Ability to configure the access banner;*
- *Ability to configure the session inactivity time before session termination or locking;*
- *Ability to update the TOE, and to verify the updates using [selection: <u>digital signature, hash comparison</u>] capability prior to installing those updates;*
- *Ability to configure the authentication failure parameters for FIA_AFL.1;*
- *Ability to configure firewall rules;*

- *[selection:*
  - *Ability to configure audit behaviour;*
  - *Ability to configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1;*
  - *Ability to configure the cryptographic functionality;*
  - *Ability to configure thresholds for SSH rekeying;*
  - *Ability to configure the lifetime for IPsec SAs;*
  - *Ability to configure the interaction between TOE components;*
  - *Ability to re-enable an Administrator account;*
  - *Ability to set the time which is used for time-stamps;*
  - *Ability to configure the reference identifier for the peer;*
  - *No other capabilities.]*

]

Application Note 24

The TOE must provide functionality for both local and remote administration in general. This cPP does not mandate, though, a specific security management function to be available either through the local administration interface, the remote administration interface or both. The TSS shall detail which security management functions are available through which interface(s). The TOE must provide functionality to configure the access banner for FTA_TAB.1 and the session inactivity time(s) for FTA_SSL_EXT.1 and FTA_SSL.3. The item "Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates" includes the relevant management functions from FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_X509_EXT.2.2 and FPT_TUD_EXT.1.2 and FPT_TUD_EXT.2.2 (if included in the ST and if they include an Administrator-configurable action). Similarly, the selection "Ability to configure audit behaviour" includes the relevant management functions from FMT_MOF.1/Services and FMT_MOF.1/Functions, (for all of these SFRs that are included in the ST). If the TOE offers the ability for a remote Administrator account to be disabled in line with FIA_AFL.1 them the ST author should select "Ability to re-enable an Administrator account" to allow the account to be re-enabled by a local Administrator. If the TOE offers the ability for the Administrator to configure the audit behaviour, configure the services available prior to identification or authentication, or if any of the cryptographic functionality on the TOE can be configured, or if the ST is describing a distributed TOE, then the ST author makes the appropriate choice or choices in the second selection, otherwise select "No other capabilities" (in the latter case the selection may alternatively be left blank in the ST).

The selection "Ability to configure thresholds for SSH rekeying" shall be included in the ST if the TOE supports configuration of the thresholds for the mechanisms used to fulfil FCS_SSHC_EXT.1.8 or FCS_SSHS_EXT.1.8 (such configuration then requires the inclusion of FMT_MOF.1/Functions in the ST). If the TOE places limits on the values accepted for the thresholds, then this is stated in the TSS.

The selection "Ability to configure lifetime for IPsec SAs" shall be included in the ST if the TOE supports secure communication via IPsec and the FCS_IPSEC_EXT.1 requirements are included in the ST. The configuration of the lifetime for IPsec SAs needs to be in line with the selection in FCS_IPSEC_EXT.1.7 (such configuration then requires the inclusion of FMT_MOF.1/Functions in the ST).

The selection "Ability to set the time which is used for time-stamps" shall be included in the ST if the TOE allows the Administrator to set the time of the device which is then used in time stamps. This option shall not be selected if the TOE does not allow manual time setting but only relies on synchronization with external time sources like NTP servers.

The selection "Ability to configure the reference identifier for the peer" shall be included in the ST if the TOE supports secure communications via the IPsec protocol and the FCS_IPSEC_EXT.1 requirements are included in the ST. For TOEs that support only IP address and FQDN identifier types, configuration of the reference identifier may be the same as configuration of the peer's name for the purposes of connection.

For distributed TOEs the interaction between TOE components will be configurable (see FCO_CPC_EXT.1). Therefore, the ST author includes the selection "Ability to configure the interaction between TOE components" for distributed TOEs. A simple example would be the change of communication protocol according to FPT_ITT.1. Another example would be changing the management of a TOE component from direct remote administration to remote administration through another TOE component. A more complex use case would be if the realization of an SFR is achieved through two or more TOE components and the responsibilities between the two or more components could be modified.

For distributed TOEs that implement a registration channel (as described in FCO_CPC_EXT.1.2), the ST author uses the selection "Ability to configure the cryptographic functionality" in this SFR, and its corresponding mapping in the TSS, to describe the configuration of any cryptographic aspects of the registration channel that can be modified by the operational environment in order to improve the channel security (cf. the description of the content of Preparative Procedures in [SD-ND, 3.6.1.2]).

6.7.4 Security management roles (FMT_SMR)

6.7.4.1 FMT_SMR.2 Restrictions on security roles

FMT_SMR.2	Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

- **Security Administrator**.

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 Security Administrator role shall be able to administer the TOE locally;**
- **The Security Administrator role shall be able to administer the TOE remotely**

are satisfied.

Application Note 25

FMT_SMR.2.3 requires that a Security Administrator be able to administer the TOE through the local console and through a remote mechanism. The ST Author must select FTP_ITC.1, FPT_ITT.1 and/or FTP_TRP.1/Admin to demonstrate how secure communication is achieved.

For distributed TOEs not every TOE component is required to implement its own user management to fulfil this SFR. At least one component has to support authentication and identification of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. For the other TOE components authentication as Security Administrator can be realized through the use of a trusted channel (either according to FTP_ITC.1 or FPT_ITT.1) from a component that supports the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. The identification of users according to FIA_UIA_EXT.1.2 and the association of users with roles according to FMT_SMR.2.2 is done through the components that support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. TOE components that authenticate Security Administrators through the use of a trusted channel are not required to support local administration of the component as defined in FMT_SMR.2.3.

6.8 Protection of the TSF (FPT)

This section defines requirements for the TOE to protect critical security data such as keys and passwords, to provide self-tests that monitor continued correct operation of the TOE (including detection of failures of firmware or software integrity), and to provide trusted methods for updates to the TOE firmware/software. In addition, the TOE is required to provide reliable timestamps in order to support accurate audit recording under the FAU_GEN family.

6.8.1 Protection of TSF Data (Extended – FPT_SKP_EXT)

6.8.1.1 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)

FPT_SKP_EXT.1	**Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)**

FPT_SKP_EXT.1.1 The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

Application Note 26

The intent of this requirement is for the device to protect keys, key material, and authentication credentials from unauthorized disclosure. This data should only be accessed for the purposes of their assigned security functionality, and there is no need for them to be displayed/accessed at any other time. This requirement does not prevent the device from providing indication that these exist, are in use, or are still valid. It does, however, restrict the reading of the values outright.

6.8.2 Protection of Administrator Passwords (Extended – FPT_APW_EXT)

6.8.2.1 FPT_APW_EXT.1 Protection of Administrator Passwords

FPT_APW_EXT.1	**Protection of Administrator Passwords**

FPT_APW_EXT.1.1 The TSF shall store passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext passwords.

Application Note 27

The intent of the requirement is that raw password authentication data is not stored in the clear, and that no user or Administrator is able to read the plaintext password through "normal" interfaces. An all-powerful Administrator could directly read memory to capture a password but is trusted not to do so. Passwords should be obscured during entry on the local console in accordance with FIA_UAU.7.

6.8.3 TSF testing (Extended – FPT_TST_EXT)

In order to detect some number of failures of underlying security mechanisms used by the TSF, the TSF will perform self-tests. The extent of this self-testing is left to the product developer, but a more comprehensive set of self-tests should result in a more trustworthy platform on which to develop enterprise architecture.

(For this component, selection-based requirements exist in Appendix B)

6.8.3.1 FPT_TST_EXT.1 TSF Testing (Extended)

FPT_TST_EXT.1	**TSF testing**

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [selection: during initial start-up (on power on), periodically during normal operation, at the request of the authorised user, at the conditions [*assignment: conditions under which self-tests should occur*]] to demonstrate the correct operation of the TSF: [*assignment: list of self-tests run by the TSF*].

Application Note 28

It is expected that self-tests are carried out during initial start-up (on power on). Other options should only be used if the developer can justify why they are not carried out during initial start-up. It is expected that at least self-tests for verification of the integrity of the firmware and software as well as for the correct operation of cryptographic functions necessary to fulfil the SFRs will be performed. If not all self-tests are performed during start-up multiple iterations of this SFR are used with the appropriate options selected. In future versions of this cPP the suite of self-tests will be required to contain at least mechanisms for measured boot including self-tests of the components which perform the measurement.

Non-distributed TOEs may internally consist of several components that contribute to enforcing SFRs. Self-testing shall cover all components that contribute to enforcing SFRs and verification of integrity shall cover all software that contributes to enforcing SFRs on all components.

For distributed TOEs all TOE components have to perform self-tests. This does not necessarily mean that each TOE component has to carry out the same self-tests: the ST describes the applicability of the selection (i.e. when self-tests are run) and the final assignment (i.e. which self-tests are carried out) to each TOE component.

Application Note 29

If certificates are used by the self-test mechanism (e.g. for verification of signatures for integrity verification), certificates are validated in accordance with FIA_X509_EXT.1/Rev and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TST_EXT.2 must be included in the ST.

6.8.4 Trusted Update (FPT_TUD_EXT)

Failure by the Security Administrator to verify that updates to the system can be trusted may lead to compromise of the entire system. To establish trust in the source of the updates, the system can provide cryptographic mechanisms and procedures to procure the update, check the update cryptographically through the TOE-provided digital signature mechanism, and install the update on the system. While there is no requirement that this process be completely automated, guidance documentation will detail any procedures that must be performed manually, as well as the manner in which the Administrator ensures that the signature on the update is valid.

(For this family, selection-based requirements exist in Appendix B)

6.8.4.1 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1	**Trusted update**

FPT_TUD_EXT.1.1 The TSF shall provide Security Administrators the ability to query the currently executing version of the TOE firmware/software and [selection: the most recently installed version of the TOE firmware/software; no other TOE firmware/software version].

Application Note 30

If a trusted update can be installed on the TOE with a delayed activation the version of both the currently executing image and the installed but inactive image must be provided. In this case the option 'the most recently installed version of the TOE firmware/software' needs to be chosen from the selection in FPT_TUD_EXT.1.1 and the TSS needs to describe how and when the inactive version becomes active. If all trusted updates become active as part of the installation process, only the currently executing version needs to be provided. In this case the option 'no other TOE firmware/software version' shall be chosen from the selection in FPT_TUD_EXT.1.1..

For a distributed TOE, the method of determining the installed versions on each component of the TOE is described in the operational guidance.

FPT_TUD_EXT.1.2 The TSF shall provide Security Administrators the ability to manually initiate updates to TOE firmware/software and [selection: support automatic checking for updates, support automatic updates, no other update mechanism].

Application Note 31

The selection in FPT_TUD_EXT.1.2 distinguishes the support of automatic checking for updates and support of automatic updates. The first option refers to a TOE that checks whether a new update is available, communicates this to the Administrator (e.g. through a message during an administrative session, through log files) but requires some action by the

Administrator to actually perform the update. The second option refers to a TOE that checks for updates and automatically installs them upon availability.

The TSS explains what actions are involved in the TOE support when using the "support automatic checking for updates" or "support automatic updates" selections.

When published hash values (see FPT_TUD_EXT.1.3) are used to protect the trusted update mechanism, the TOE must not automatically download the update file(s) together with the hash value (either integrated in the update file(s) or separately) and automatically install the update without any active authorization by the Security Administrator, even when the calculated hash value matches the published hash value. When using published hash values to protect the trusted update mechanism, the option "support of automatic updates" must not be used (automated checking for updates is permitted, though). The TOE may automatically download the update file(s) themselves but not to the hash value. For the published hash approach, it is intended that a Security Administrator is always required to give active authorisation for installation of an update (as described in more detail under FPT_TUD_EXT.1.3) below. Due to this, the type of update mechanism is regarded as "manually initiated update", even if the update file(s) may be downloaded automatically. A fully automated approach (without Security Administrator intervention) can only be used when "digital signature mechanism" is selected in FPT_TUD_EXT.1.3 below.

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [selection: digital signature mechanism, published hash] prior to installing those updates.

Application Note 32

The digital signature mechanism referenced in the selection of FPT_TUD_EXT.1.3 is one of the algorithms specified in FCS_COP.1/SigGen. The published hash referenced in FPT_TUD_EXT.1.3 is generated by one of the functions specified in FCS_COP.1/Hash. The ST author should choose the mechanism implemented by the TOE; it is acceptable to implement both mechanisms.

When published hash values are used to secure the trusted update mechanism, an active authorization of the update process by the Security Administrator is always required. The secure transmission of an authentic hash value from the developer to the Security Administrator is one of the key factors to protect the trusted update mechanism when using published hashes and the guidance documentation needs to describe how this transfer has to be performed. For the verification of the trusted hash value by the Security Administrator different use cases are possible. The Security Administrator could obtain the published hash value as well as the update file(s) and perform the verification outside the TOE while the hashing of the update file(s) could be done by the TOE or by other means. Authentication as Security Administrator and initiation of the trusted update would in this case be regarded as "active authorization" of the trusted update. Alternatively, the Administrator could provide the TOE with the published hash value together with the update file(s) and the hashing and hash comparison is performed by the TOE. In case of successful hash verification, the TOE can perform the update without any additional step by the Security Administrator. Authentication as Security Administrator and sending the hash value to the TOE is regarded as "active authorization" of the trusted update (in case of successful hash verification), because the Administrator is expected to load the hash value only to the TOE when intending to perform the update. As long as the transfer of the hash value to the TOE is performed by the Security

Administrator, loading of the update file(s) can be performed by the Security Administrator or can be automatically downloaded by the TOE from a repository.

If the digital signature mechanism is selected, the verification of the signature shall be performed by the TOE itself. For the published hash option, the verification can be done by the TOE itself as well as by the Security Administrator. In the latter case use of TOE functionality for the verification is not mandated, so verification could be done using non-TOE functionality of the device containing the TOE or without using the device containing the TOE.

For distributed TOEs all TOE components shall support Trusted Update. The verification of the signature or hash on the update shall either be done by each TOE component itself (signature verification) or for each TOE component (hash verification).

Updating a distributed TOE might lead to the situation where different TOE components are running different software versions. Depending on the differences between the different software versions the impact of a mixture of different software versions might be no problem at all or critical to the proper functioning of the TOE. The TSS shall detail the mechanisms that support the continuous proper functioning of the TOE during trusted update of distributed TOEs.

Application Note 33

Future versions of this cPP will mandate the use of a digital signature mechanism for trusted updates.

Application Note 34

If certificates are used by the update verification mechanism, certificates are validated in accordance with FIA_X509_EXT.1/Rev and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TUD_EXT.2 must be included in the ST.

Application Note 35

"Update" in the context of this SFR refers to the process of replacing a non-volatile, system resident software component with another. The former is referred to as the NV image, and the latter is the update image. While the update image is typically newer than the NV image, this is not a requirement. There are legitimate cases where the system owner may want to rollback a component to an older version (e.g. when the component manufacturer releases a faulty update, or when the system relies on an undocumented feature no longer present in the update). Likewise, the owner may want to update with the same version as the NV image to recover from faulty storage.

All discrete firmware and software elements (e.g. applications, drivers, and kernel) of the TSF need to be protected, i.e. they should either be digitally signed by the corresponding manufacturer and subsequently verified by the mechanism performing the update or a hash should be published for them which needs to be verified before the update.

6.8.5 Time stamps (Extended – FPT_STM_EXT))

6.8.5.1 FPT_STM.EXT.1 Reliable Time Stamps

FPT_STM.EXT.1	Reliable Time Stamps

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [selection: allow the Security Administrator to set the time, synchronise time with external time sources].

Application Note 36

Reliable time stamps are expected to be used with other TSF, e.g. for the generation of audit data to allow the Security Administrator to investigate incidents by checking the order of events and to determine the actual local time when events occurred. The decision about the required level of accuracy of that information is up to the Administrator. The TOE depends on external time and date information, either provided manually by the Security Administrator or through the use of one or more external time sources like NTP servers. The corresponding option(s) shall be chosen from the selection in FPT_STM_EXT.1.2. The use of a local real-time clock and the automatic synchronisation with an external time source (e.g. NTP server) is recommended but not mandated. Note that for the communication with an external time source like an NTP server, the use of FTP_ITC.1 is optional but not mandated. The ST author describes in the TSS how the external time and date information is received by the TOE and how this information is maintained.

The term "reliable time stamps" refers to the strict use of the time and date information, that is provided externally, and the logging of all discontinuous changes to the time settings including information about the old and new time. With this information the real time for all audit data can be determined. Note, that all discontinuous time changes, Administrator actuated or changed via an automated process, must be audited. No audit is needed when time is changed via use of kernel or system facilities – such as daytime (3) – that exhibit no discontinuities in time.

For distributed TOEs it is expected that the Security Administrator ensures synchronization between the time settings of different TOE components. All TOE components shall either be in sync (e.g. through synchronisation between TOE components or through synchronisation of different TOE components with external NTP servers) or the offset should be known to the Administrator for every pair of TOE components. This includes TOE components synchronized to different time zones.

6.9 TOE Access (FTA)

This section specifies requirements associated with security of administrative sessions carried out on the TOE. In particular, both local and remote sessions are monitored for inactivity and either locked or terminated when a threshold time period is reached. Administrators must also be able to positively terminate their own interactive sessions, and must have an advisory notice displayed at the start of each session.

6.9.1 TSF-initiated Session Locking (Extended – FTA_SSL_EXT)

6.9.1.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1	TSF-initiated Session Locking

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [selection:

- lock the session - disable any activity of the user's data access/display devices other than unlocking the session, and requiring that the Administrator re-authenticate to the TSF prior to unlocking the session;
- terminate the session]

after a Security Administrator-specified time period of inactivity.

6.9.2 Session locking and termination (FTA_SSL)

6.9.2.1 FTA_SSL.3 TSF-initiated Termination (Refinement)

FTA_SSL.3	TSF-initiated Termination

FTA_SSL.3.1: The TSF shall terminate **a remote** interactive session after a **Security Administrator-configurable time interval of session inactivity**.

6.9.2.2 FTA_SSL.4 User-initiated Termination (Refinement)

FTA_SSL.4	User-initiated Termination

FTA_SSL.4.1: The TSF shall allow **Administrator**-initiated termination of the **Administrator's** own interactive session.

6.9.3 TOE access banners (FTA_TAB)

6.9.3.1 FTA_TAB.1 Default TOE Access Banners (Refinement)

FTA_TAB.1	Default TOE Access Banners

FTA_TAB.1.1: Before establishing **an administrative user** session the TSF shall display **a Security Administrator-specified** advisory **notice and consent** warning message regarding use of the TOE.

Application Note 37

This requirement is intended to apply to interactive sessions between a human user and a TOE. IT entities establishing connections or programmatic connections (e.g., remote procedure calls over a network) are not required to be covered by this requirement.

6.10 Trusted path/channels (FTP)

To address the issues concerning transmitting sensitive data to and from the TOE, compliant TOEs will provide encryption for these communication paths between themselves and the endpoint. These channels are implemented using one (or more) of five standard protocols: IPsec, TLS, DTLS, HTTPS, and SSH. These protocols are specified by RFCs that offer a variety of implementation choices. Requirements have been imposed on some of these choices (particularly those for cryptographic primitives) to provide interoperability and resistance to cryptographic attack.

In addition to providing protection from disclosure (and detection of modification) for the communications, each of the protocols described (IPsec, SSH, TLS, DTLS and HTTPS) offer two-way authentication of each endpoint in a cryptographically secure manner, meaning that even if there was a malicious attacker between the two endpoints, any attempt to represent themselves to either endpoint of the communications path as the other communicating party would be detected.

6.10.1 Trusted Channel (FTP_ITC)

6.10.1.1 FTP_ITC.1 Inter-TSF trusted channel (Refinement)

FTP_ITC.1	Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall **be capable of using [selection: IPsec, SSH, TLS, DTLS, HTTPS] to provide** a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server, [selection: authentication server, *assignment: [other capabilities]*, no other capabilities]** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

FTP_ITC.1.2 The TSF shall permit **the TSF or the authorized IT entities** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [*assignment: list of* ***services for which the TSF is able to initiate communications***].

Application Note 38

The intent of the above requirement is to provide a means by which a cryptographic protocol may be used to protect external communications with authorized IT entities that the TOE interacts with to perform its functions. The TOE uses at least one of the listed protocols for communications with the server that collects the audit information. If it communicates with an authentication server (e.g., RADIUS), then the ST author chooses "authentication server" in FTP_ITC.1.1 and this connection must be capable of being protected by one of the listed protocols. If other authorized IT entities are protected, the ST author makes the appropriate assignments (for those entities) and selections (for the protocols that are used to protect those connections). The ST author selects the mechanism or mechanisms supported by the TOE, and then ensures that the detailed protocol requirements in Appendix B corresponding to their selection are included in the ST.

While there are no requirements on the party initiating the communication, the ST author lists in the assignment for FTP_ITC.1.3 the services for which the TOE can initiate the communication with the authorized IT entity.

The requirement implies that not only are communications protected when they are initially established, but also on resumption after an outage. It may be the case that some part of the TOE setup involves manually setting up tunnels to protect other communication, and if after an outage the TOE attempts to re-establish the communication automatically with (the necessary) manual intervention, there may be a window created where an attacker might be able to gain critical information or compromise a connection.

Where public key certificates are used in support of an FTP_ITC.1 channel, FIA_X509_EXT.1/Rev is to be used (this requires checking certificate revocation), and not the iteration FIA_X509_EXT.1/ITT which is only for use in inter-component channels of a distributed TOE.

6.10.2 Trusted Path (FTP_TRP)

6.10.2.1 FTP_TRP.1/Admin Trusted Path (Refinement)

FTP_TRP.1/Admin	**Trusted Path**

FTP_TRP.1.1/Admin The TSF shall **be capable of using [selection: <u>DTLS, 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 **disclosure and provides detection of modification of the channel data.**

FTP_TRP.1.2/Admin The TSF shall permit **remote Administrators** to initiate communication via the trusted path.

FTP_TRP.1.3/Admin The TSF shall require the use of the trusted path for **initial Administrator authentication and all remote administration actions**.

Application Note 39

This requirement ensures that authorized remote Administrators initiate all communication with the TOE via a trusted path, and that all communication with the TOE by remote Administrators is performed over this path. The data passed in this trusted communication channel is encrypted as defined by the protocol chosen in the first selection. The ST author selects the mechanism or mechanisms supported by the TOE, and then ensures that the detailed protocol requirements in Appendix B corresponding to their selection are included in the ST.

6.11 Firewall (FFW)

6.11.1 Stateful Traffic Filter Firewall (FFW_RUL_EXT)

To address the issues associated with unauthorized disclosure of information, inappropriate access to services, misuse of services, disruption or denial of services, and network-based reconnaissance, compliant TOE's will implement a Stateful Traffic Filtering capability. That capability will restrict the flow of network traffic between protected networks and other attached networks based on network addresses and ports of the network nodes originating (source) and/or receiving (destination) applicable network traffic as well as on established connection information.

Stateful packet inspection is used to aid in the performance of packet flow through the TOE. Rather than apply the ruleset against each packet that is processed at a TOE interface, the TOE will determine whether a packet belongs to an "approved" established connection. The minimum set of attributes that are used to determine whether a packet is part of an established session are mandated for TCP and UDP, and the ST author is allowed to expand the attributes considered for TCP sessions, and add the ICMP protocol if they desire.

Compliant TOEs will implement the ability to log the flow of network traffic. Specifically, the TOE will provide the means for administrators to configure firewall specific firewall rules to 'log' when network traffic is found to match the configured rule. As a result, matching a firewall rule configured to 'log' will result in informative event logs whenever a match occurs.

6.11.1.1 FFW_RUL_EXT.1 Stateful Traffic Filtering

FFW_RUL_EXT.1	**Stateful Traffic Filtering**

FFW_RUL_EXT.1.1 The TSF shall perform Stateful Traffic Filtering on network packets processed by the TOE.

Application Note 40

This element identifies the policy (Stateful Traffic Filtering) that is applied to the network packets that are processed at the TOE's interfaces. Every packet that is received at a TOE's interface either has the ruleset that expresses this policy applied, or it is determined that the packet belongs to an established connection. The remaining elements in this component provide the details of the policy.

This requirement is to be enforced even if the network interface is saturated/overwhelmed with network traffic.

It is important to note that the TOE, which also includes the underlying platform, cannot permit network packets to flow unless the ruleset contains a rule that permits the flow, or the packet is deemed to belong to an established connection that has been permitted to flow. This principle must hold true during TOE startup, and upon failures the TOE may encounter.

FFW_RUL_EXT.1.2 The TSF shall allow the definition of Stateful Traffic Filtering rules using the following network protocol fields:

- ICMPv4
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source address
  - Destination Address
  - Transport Layer Protocol
- IPv6

  - o Source address
  - o Destination Address
  - o Transport Layer Protocol
  - o [selection: *IPv6 Extension header type [assignment: list of fields in IPv6 extension header], no other field*]
- TCP
  - o Source Port
  - o Destination Port
- UDP
  - o Source Port
  - o Destination Port
- and distinct interface.

Application Note 41

This element identifies the various attributes that are applicable when constructing rules to be enforced by this requirement – the applicable interface is a property of the TOE and the rest of the identified attributes are defined in the associated RFCs. Note that the 'Transport Layer Protocol' is the IPv4/IPv6 field that identifies the applicable protocol, such as TCP, UDP, ICMP, or GRE. IPv6 extension headers are defined in RFC 2460 and the ST author may specify which fields within each supported extension header, if any may be used as attributes in the construction of an inspection rule. Also, 'Interface' identified above is the external port where the applicable network traffic was received or will be sent.

FFW_RUL_EXT.1.3 The TSF shall allow the following operations to be associated with Stateful Traffic Filtering rules: permit or drop with the capability to log the operation.

Application Note 42

This element defines the operations that can be associated with rules used to match network traffic. Note that the data to be logged is identified in the Security Audit requirements in Table 2.

FFW_RUL_EXT.1.4 The TSF shall allow the Stateful Traffic Filtering rules to be assigned to each distinct network interface.

Application Note 43

This element identifies where rules can be assigned. Specifically, a conforming TOE must be able to assign filtering rules to each of its available and distinct network interfaces that handle layer 3 and 4 network traffic. A distinct network interface can be physical or logical but it does not necessarily required to be visible from the network perspective (e.g. it does not need to have an IP address assigned to it).

Note that there could be a separate ruleset for each interface or alternately a shared ruleset that somehow associates rules with specific interfaces.

FFW_RUL_EXT.1.5 The TSF shall:

a) accept a network packet without further processing of Stateful Traffic Filtering rules if it matches an allowed established session for the following protocols: TCP, UDP, [selection: *ICMP, no other protocols*] based on the following network packet attributes:

1. TCP: source and destination addresses, source and destination ports, sequence number, Flags;
2. UDP: source and destination addresses, source and destination ports;
3. [selection: *'ICMP: source and destination addresses, type, [selection: code, [assignment: list of matching attributes]]', no other protocols*].

b) Remove existing traffic flows from the set of established traffic flows based on the following: [selection: *session inactivity timeout, completion of the expected information flow*].

Application Note 44

This element requires that the protocols be identified for which the TOE can determine and manage the state such that sessions can be established and are used to make traffic flow decisions as opposed to fully processing the configured rules. This element also requires that applicable attributes used to determine whether a network packet matches and established session are identified.

If ICMP is selected as a protocol the source and destination addresses are required to be considered when determining if a packet belongs to an established "connection". The type and code attributes may be used to provide a more robust capability in determining whether an ICMP packet is what is expected in an established connection flow. For example, one would not expect echo replies to be part of a flow if an echo request had not been received. The open assignment in the selection for ICMP attributes is left for implementations that may use IPv6 attributes.

Item b) in this element requires specification of how the firewall can determine that established information flows should be removed from the set of established information flows by observing events such as the termination of a TCP session initiated by either endpoint with FIN flags in the TCP packet. If protocols are handled differently, it is expected that the ST would identify those differences.

FFW_RUL_EXT.1.6 The TSF shall enforce the following default Stateful Traffic Filtering rules on all network traffic:

a) The TSF shall drop and be capable of [selection: *counting, logging*] packets which are invalid fragments;
b) The TSF shall drop and be capable of [selection: *counting, logging*] fragmented packets which cannot be re-assembled completely;
c) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a broadcast network;
d) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a multicast network;The TSF shall drop and be capable of logging network packets where the source address of the network packet is defined as being a loopback address;
e) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address "reserved for future use" (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4;
f) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as an "unspecified address" or an address "reserved for future definition and use" (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;

g) The TSF shall drop and be capable of logging network packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified; and
h) [selection: *[assignment: other default rules enforced by the TOE], no other rules*].

Application Note 45

Future revisions of this cPP will require that the TOE implements these default rules without the need to apply configuration.

FFW_RUL_EXT.1.7 The TSF shall be capable of dropping and logging according to the following rules:

a) The TSF shall drop and be capable of logging network packets where the source address of the network packet is equal to the address of the network interface where the network packet was received;
b) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is a link-local address;
c) The TSF shall drop and be capable of logging network packets where the source address of the network packet does not belong to the networks associated with the network interface where the network packet was received.

Application Note 46

Note that these rules may be configured.

FFW_RUL_EXT.1.8 , The TSF shall process the applicable Stateful Traffic Filtering rules in an administratively defined order.

Application Note 47

This element requires that an administrator is able to define the order in which configured filtering rules are processed for matches. The filtering rules are only applicable when an allowed session has not been established or a dynamic rule has been created.

FFW_RUL_EXT.1.9 The TSF shall deny packet flow if a matching rule is not identified.

Application Note 48

This element requires that, except when a packet is part of an established session, the behavior is always to deny network traffic when no rules apply and no other operations are required, though they are not necessarily prohibited.

FFW_RUL_EXT.1.10 The TSF shall be capable of limiting an administratively defined number of *half-open TCP connections. In the event that the configured limit is reached, new connection attempts shall be dropped and the drop event shall be [selection: counted, logged]*.

Application Note 49

A half-open TCP connection is one that has not completed the full three-way handshake as defined in RFC 793. Incomplete TCP connections i.e. those that have completed the SYN and SYN-ACK portions of the three-way handshake consume valuable resources in end hosts and stateful traffic filtering devices in the traffic path and, in sufficient volume, can lead to a denial of service condition. To protect itself, and any targeted protected services, compliant TOEs shall be capable of limiting the number of half-open TCP connections.

7. Security Assurance Requirements

This cPP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.

This section lists the set of SARs from CC part 3 that are required in evaluations against this cPP. Individual Evaluation Activities to be performed are specified in [SD-FW] and [SD-ND].

The general model for evaluation of TOEs against STs written to conform to this cPP is as follows: after the ST has been approved for evaluation, the ITSEF will obtain the TOE, supporting environmental IT (if required), and the guidance documentation for the TOE. The ITSEF is expected to perform actions mandated by the Common Evaluation Methodology (CEM) for the ASE and ALC SARs. The ITSEF also performs the Evaluation Activities contained within the SD, which are intended to be an interpretation of the other CEM assurance requirements as they apply to the specific technology instantiated in the TOE. The Evaluation Activities that are captured in [SD-FW] and [SD-ND] also provide clarification as to what the developer needs to provide to demonstrate the TOE is compliant with the cPP.

The TOE security assurance requirements are identified in Table 3.

Assurance Class	**Assurance Components**
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Stated security requirements (ASE_REQ.1)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Labelling of the TOE (ALC_CMC.1)
	TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing – conformance (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

Table 3: Security Assurance Requirements

7.1 ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM. In addition, there may be Evaluation Activities specified within [SD-FW] and [SD-ND] that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

Appendix D provides a description of the information expected to be provided regarding the quality of entropy in the random bit generator.

ASE_TSS.1.1C Refinement: The TOE summary specification shall describe how the TOE meets each SFR. **In the case of entropy analysis, the TSS is used in conjunction with required supplementary information on Entropy.**

The requirements for exact conformance of the Security Target are described in section 2.

7.2 ADV: Development

The design information about the TOE is contained in the guidance documentation available to the end user as well as the TSS portion of the ST, and any required supplementary information required by this cPP that is not to be made public.

7.2.1 Basic Functional Specification (ADV_FSP.1)

The functional specification describes the TOE Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this cPP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE users, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional "functional specification" documentation is necessary to satisfy the Evaluation Activities specified in [SD-ND] and [SD-FW].

The Evaluation Activities in [SD-ND] are associated with the applicable SFRs; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

7.3 AGD: Guidance Documentation

The guidance documents will be provided with the ST. Guidance must include a description of how the IT personnel verifies that the Operational Environment can fulfil its role for the security functionality. The documentation should be in an informal style and readable by the IT personnel.

Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes:

- instructions to successfully install the TSF in that environment; and
- instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability.

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the Evaluation Activities specified in [SD-FW] and [SD-ND].

7.3.1 Operational User Guidance (AGD_OPE.1)

The operational user guidance does not have to be contained in a single document. Guidance to users, Administrators and application developers can be spread among documents or web pages.

The developer should review the Evaluation Activities contained in [SD-FW] and [SD-ND] to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

7.3.2 Preparative Procedures (AGD_PRE.1)

As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

It is noted that specific requirements for Preparative Procedures are defined in [SD-ND] for distributed TOEs as part of the Evaluation Activities for FCO_CPC_EXT.1 and FTP_TRP.1/Join.

7.4 Class ALC: Life-cycle Support

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

7.4.1 Labelling of the TOE (ALC_CMC.1)

This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a "hard label" (e.g., stamped into the metal, paper label) or a "soft label" (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

7.4.2 TOE CM Coverage (ALC_CMS.1)

Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

7.5 Class ATE: Tests

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

7.5.1 Independent Testing – Conformance (ATE_IND.1)

Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes "evaluated configuration" instructions). The focus of the testing is to confirm that the requirements specified in Section 5.1.6 are being met. The Evaluation Activities in [SD-ND] and [SD-FW] identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

7.6 Class AVA: Vulnerability Assessment

For the first generation of this cPP, the iTC is expected to survey open sources to discover what vulnerabilities have been discovered in these types of products and provide that content into the AVA_VAN discussion. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. This information will be used in the development of future protection profiles.

7.6.1 Vulnerability Survey (AVA_VAN.1)

[SD-ND, Appendix A] provides a guide to the evaluator in performing a vulnerability analysis.

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 types of requirements specified in Appendices A and B.

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

If a TOE fulfils any of the optional requirements, the vendor is encouraged to add the related functionality to the ST. Therefore, in the application notes of this chapter the wording "This option should be chosen..." is repeatedly used. But it also is used to emphasize that this option should only be chosen if the TOE provides the related functionality and that it is not necessary to implement the related functionality to be compliant to the cPP. ST authors are free to choose none, some or all SFRs defined in this chapter. Just the fact that a product supports a certain functionality does not mandate to add any SFR defined in this chapter.

A.1 Audit Events for Optional SFRs

Requirement	Auditable Events	Additional Audit Record Contents
FAU_STG.1	None.	None.
FAU_STG_EXT.2/LocSpace	None.	None.
FAU_STG.3/LocSpace	Low storage space for audit events.	None.
FIA_X509_EXT.1/ITT	Unsuccessful attempt to validate a certificate	Reason for failure
FMT_MOF.1/Services	Starting and stopping of services.	None.
FMT_MTD.1/CryptoKeys	Management of cryptographic keys.	None.

FPT_ITT.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Join	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.
FCO_CPC_EXT.1	Enabling communications between a pair of components. Disabling communications between a pair of components.	Identities of the endpoints pairs enabled or disabled.
FFW_RUL_EXT.2	To be defined in ST.	To be defined in ST.

Table 4: TOE Optional SFRs and Auditable Events

Application Note 50

The audit event for FIA_X509_EXT.1/ITT is based on the TOE not being able to complete the certificate validation by ensuring the following:

- *the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.*
- *Verification of the digital signature of the trusted hierarchical CA*
- *read/access the CRL or access the OCSP server (according to selection in the ST).*

If any of these checks fails, then an audit event with the failure should be written to the audit log.

A.2 Security Audit (FAU)

A.2.1 Security audit event storage (FAU_STG.1 & Extended – FAU_STG_EXT)

Local storage space for audit data may be necessary on the TOE itself, and the TOE may then claim protection of the audit trail against unauthorised modification (including deletion) as described in FAU_STG.1. The local storage space for audit data of a network device is also limited, and if the local storage space is exceeded then audit data might be lost. A security Administrator might be interested in the number of dropped, overwritten, etc. audit records. This number might serve as an indication if a severe problem has occurred after the storage space was exceeded that continuously generated audit data. Therefore, FAU_STG_EXT.2/LocSpace and FAU_STG.3/LocSpace are defined to express these optional capabilities of a network device.

A.2.1.1 FAU_STG.1 Protected audit trail storage

FAU_STG.1	Protected audit trail storage

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2 The TSF shall be able to prevent unauthorised modifications to the stored audit records in the audit trail.

A.2.1.2 FAU_ STG_EXT.2/LocSpace Counting lost audit data

FAU_STG_EXT.2/LocSpace	Counting lost audit data

FAU_STG_EXT.2.1/LocSpace The TSF shall provide information about the number of [selection: dropped, overwritten, *assignment: other information*] audit records in the case where the local storage has been filled and the TSF takes one of the actions defined in FAU_STG_EXT.1.3.

Application Note 51

This option should be chosen if the TOE supports this functionality.

In case the local storage for audit records is cleared by the Administrator, the counters associated with the selection in the SFR should be reset to their initial value (most likely to 0). The guidance documentation should contain a warning for the Administrator about the loss of audit data when he clears the local storage for audit records.

For distributed TOEs each component that implements counting of lost audit data has to provide a mechanism for Administrator access to, and management of, this information.

If FAU_STG_EXT.2/LocSpace is added to the ST, the ST has to make clear any situations in which lost audit data is not counted.

A.2.1.3 FAU_ STG.3/LocSpace Action in case of possible audit data loss

FAU_STG.3/LocSpace	**Action in case of possible audit data loss**

FAU_STG.3.1/LocSpace The TSF shall generate a warning to inform the Administrator if the audit trail exceeds the local audit trail storage capacity.

Application Note 52

This option should be chosen if the TOE generates a warning to inform the Administrator before the local storage space for audit data is used up. This might be useful if auditable events are stored on local storage space only.

It has to be ensured that the warning message required by FAU_STG.3.1/LocSpace can be communicated to the Administrator. The communication should be done via the audit log itself because it cannot be guaranteed that an administrative session is active at the time the event occurs.

The warning should inform the Administrator when the local space to store audit data is used up and/or the TOE will lose audit data due to insufficient local space.

For distributed TOEs that implement displaying a warning when local storage space for audit data is exhausted, it has to be described which TOE components support this feature (not necessarily all TOE components have to support this feature if selected for the overall TOE). Each component that supports this feature shall either generate a warning itself or through another component.

If FAU_STG.3/LocSpace is added to the ST, the ST has to make clear any situations in which audit records might be "invisibly lost".

A.3 Identification and Authentication (FIA)

A.3.1 Authentication using X.509 certificates (Extended – FIA_X509_EXT)

A.3.1.1 FIA_X509_EXT.1/ITT Certificate Validation

FIA_X509_EXT.1/ITT	**X.509 Certificate Validation**

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

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of two certificates.
- 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 6960, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5, no revocation method]

- The TSF shall validate the extendedKeyUsage field according to the following rules:

    - 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 53

This SFR should be chosen if the TOE is distributed and the protocol(s) selected in FPT_ITT.1 utilize X.509 certificates for peer authentication. In this case, the use of revocation list checking is optional as there are additional requirements surrounding the enabling and disabling of the ITT channel as defined in FCO_CPC_EXT.1. If revocation checking is not supported, the ST author should select no revocation method. However, if certificate revocation checking is supported, the ST author selects whether this is performed using OCSP or CRLs.

The TOE shall be capable of supporting a minimum path length of two certificates. That is, it shall support a certificate hierarchy comprising of at least a self-signed root certificate and a TOE identity certificate.

The TSS shall describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the device.

If the TOE does not support functionality that uses any of the certificate types listed in the extendedKeyUsage rules in FIA_X509_EXT.1.1 then this is stated in the TSS and the relevant part of the SFR is considered trivially satisfied. However, if the TOE does support functionality that uses certificates of any of these types then the corresponding rule must of course be satisfied as in the SFR.

FIA_X509_EXT.1.2/ITT 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.

Application Note 54

This requirement applies to certificates that are used and processed by the TSF and restricts the certificates that may be added as trusted CA certificates.

A.4 Security Management (FMT)

A.4.1 Management of functions in TSF (FMT_MOF)

A.4.1.1 FMT_MOF.1/Services Management of security functions behaviour

FMT_MOF.1/Services	**Management of security functions behaviour**

FMT_MOF.1.1/Services The TSF shall restrict the ability to enable and disable functions and services to Security Administrators.

Application Note 55

FMT_MOF.1/Services should only be chosen if the Security Administrator has the ability to start and stop services. In this case the option 'starting and stopping services' shall be chosen in the selection in FAU_GEN.1.1. The term "services" is defined as for FAU_GEN.1.1 (see related Application Notes for FAU_GEN.1.1).

A.4.2 Management of TSF data (FMT_MTD)

A.4.2.1 FMT_MTD.1/CryptoKeys Management of TSF data

FMT_MTD.1/CryptoKeys	**Management of TSF data**

FMT_MTD.1.1/CryptoKeys The TSF shall restrict the ability to manage the cryptographic keys to Security Administrators.

Application Note 56

FMT_MTD.1.1/CryptoKeys restricts management of cryptographic keys to Security Administrators. It should only be chosen if cryptographic keys can be managed (e.g. modified, deleted or generated/imported) by the Security Administrator. The identifier "CryptoKeys" has been added here to separate this iteration of FMT_MTD.1 from the mandatory iteration of FMT_MTD.1 defined in Chapter 6.6.2.1 (FMT_MTD.1/CoreData).

A.5 Protection of the TSF (FPT)

A.5.1 Internal TOE TSF data transfer (FPT_ITT)

A.5.1.1 FPT_ITT.1 Basic internal TSF data transfer protection (Refinement)

FPT_ITT.1	**Basic internal TSF data transfer protection**

FPT_ITT.1.1 The TSF shall protect TSF data from **disclosure and detect its modification** when it is transmitted between separate parts of the TOE **through the use of [selection: <u>IPsec, SSH, TLS, DTLS, HTTPS</u>]**.

Application Note 57

This requirement is only applicable to distributed TOEs, and ensures that all communications between components of the distributed TOE are protected through the use of an encrypted communications channel. The data passed in this trusted communication channel are encrypted as defined by the protocol chosen in the selection. The ST author should identify the channels and protocols used by each pair of communicating components in a distributed TOE, iterating this SFR as appropriate.

This channel may also be used as the registration channel for the registration process, as described in section 3.3 and FCO_CPC_EXT.1.2.

If TLS is selected, then the requirements to have the reference identifier established by the user (FCS_TLSC_EXT.1.2) are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the "joining" component.

A.6 Trusted Path/Channels (FTP)

A.6.1 Trusted Path (FTP_TRP)

A.6.1.1 FTP_TRP.1/Join Trusted Path (Refinement)

This iteration of FTP_TRP.1 is defined as one of the options selectable for distributed TOE component registration in FCO_CPC_EXT.1 (section A.7.1).

FTP_TRP.1/Join	**Trusted Path**

FTP_TRP.1.1/Join The TSF shall provide a communication path between itself and **a joining component** ~~[selection: remote, local] users~~ that is logically distinct from other communication paths and provides assured identification of **[selection: the TSF endpoint, both joining component and TSF endpoint]** ~~its end points~~ and protection of the communicated data from **modification [selection: and disclosure, none]**.

FTP_TRP.1.2/Join The TSF shall permit [selection: the TSF, **the joining component** ~~local users, remote users~~] to initiate communication via the trusted path.

FTP_TRP.1.3/Join The TSF shall require the use of the trusted path for **joining components to the TSF under environmental constraints identified in *reference to operational guidance***.

Application Note 58

This SFR implements one of the types of channel identified in the main selection for FCO_CPC_EXT.1.2. The "joining component" in FTP_TRP.1/Join is the IT entity that is attempting to join the distributed TOE by using the registration process.

The effect of this SFR is to require the ability for components to communicate in a secure manner while the distributed TSF is being created (or when adding components to an existing distributed TSF). When creating the TSF from the initial pair of components, either of these components may be identified as the TSF for the purposes of satisfying the meaning of "TSF" in this SFR.

The selection at the end of FTP_TRP.1.1/Join recognises that in some cases confidentiality (i.e. protection of the data from disclosure) may not be provided by the channel. The ST author distinguishes in the TSS whether in this case the TOE relies on the environment to provide confidentiality (as part of the constraints referenced in FTP_TRP.1.3/Join) or whether the registration data exchanged does not require confidentiality (in which case this assertion must be justified). If "none" is selected, then this word may be omitted in the ST to improve readability.

The assignment in FTP_TRP.1.3/Join ensures that the ST highlights any specific details needed to protect the registration environment.

Note that when the ST uses FTP_TRP.1/Join for the registration channel then this channel cannot be reused as the normal inter-component communication channel (the latter channel must meet FTP_ITC.1 or FPT_ITT.1).

Specific requirements for Preparative Procedures relating to FTP_TRP.1/Join are defined in the Evaluation Activities in [SD-ND].

A.7 Communication (FCO)

A.7.1 Communication Partner Control (FCO_CPC_EXT)

The SFR in this section defines the top-level requirement for control over the way in which components are joined together under the control of a Security Administrator to create the distributed TOE (cf. section 3.3). The SFR makes use of references to other SFRs to define the lower-level characteristics of the types of channel that may be used in the registration process.

A.7.1.1 FCO_CPC_EXT.1 Component Registration Channel Definition

FCO_CPC_EXT.1	Component Registration Channel Definition

FCO_CPC_EXT.1.1 The TSF shall require a Security Administrator to enable communications between any pair of TOE components before such communication can take place.

FCO_CPC_EXT.1.2 The TSF shall implement a registration process in which components establish and use a communications channel that uses [selection:

- A channel that meets the secure channel requirements in [selection: FTP_ITC.1, FPT_ITT.1],
- A channel that meets the secure registration channel requirements in FTP_TRP.1/ Join,
- No channel].

for at least TSF data.

FCO_CPC_EXT.1.3 The TSF shall enable a Security Administrator to disable communications between any pair of TOE components.

Application Note 59

This SFR is only applicable if the TOE is distributed and therefore has multiple components that need to communicate via an internal TSF channel. When creating the TSF from the initial pair of components, either of these components may be identified as the TSF for the purposes of satisfying the meaning of "TSF" in this SFR.

The intention of this requirement is to ensure that there is a registration process that includes a positive enablement step by an Administrator before components joining a distributed TOE can communicate with the other components of the TOE and before the new component can act as part of the TSF. The registration process may itself involve communication with the joining component: many network devices use a bespoke process for this, and the security requirements for the "registration communication" are then defined in FCO_CPC_EXT.1.2. Use of this "registration communication" channel is not deemed inconsistent with the requirement of FCO_CPC_EXT.1.1 (i.e. the registration channel can be used before the enablement step, but only in order to complete the registration process).

The channel selection (for the registration channel) in FCO_CPC_EXT.1.2 is essentially a choice between the use of a normal secure channel that is equivalent to a channel used to communicate with external IT entities (FTP_ITC.1) or existing TOE components (FPT_ITT.1), or else a separate type of channel that is specific to registration (FTP_TRP.1/Join). If the TOE does not require a communications channel for registration (e.g. because the registration is achieved entirely by configuration actions by an Administrator at each of the components) then the main selection in FCO_CPC_EXT.1.2 is completed with the "No channel" option.

If the ST author selects the FTP_ITC.1/FPT_ITT.1 channel type in the main selection in FCO_CPC_EXT.1.2 then the TSS identifies the relevant SFR iteration that specifies the channel used. If the ST author selects the FTP_TRP.1/Join channel type, then the TOE Summary Specification (possibly with support from the operational guidance) describes details of the channel and the mechanisms that it uses (and describes how the registration process ensures that the channel can only be used by the intended joiner and gatekeeper). Note that the FTP_TRP.1/Join channel type may require support from security measures in the operational environment (see the definition of FTP_TRP.1/Join for details).

If the ST author selects the FTP_ITC.1/FPT_ITT.1 channel type in the main selection in FCO_CPC_EXT.1.2 then the ST identifies the registration channel as a separate iteration of FTP_ITC.1 or FPT_ITT.1 and gives the iteration identifier (e.g. "FPT_ITT.1/Join") in an ST Application Note for FCO_CPC_EXT.1.

Note that the channel set up and used for registration may be adopted as a continuing internal communication channel (i.e. between different TOE components) provided that the channel meets the requirements of FTP_ITC.1 or FPT_ITT.1. Otherwise the registration channel is closed after use and a separate channel is used for the internal communications.

Specific requirements for Preparative Procedures relating to FCO_CPC_EXT.1 are defined in the Evaluation Activities in [SD-ND].

A.8 Firewall (FFW)

A.8.1 Stateful Traffic Filter Firewall (FFW_RUL)

A.8.1.1 FFW_RUL_EXT.2 Stateful Filtering of Dynamic Protocols

FFW_RUL_EXT.2	Sateful Filtering of Dynamic Protocols

FFW_RUL_EXT.2.1 The TSF shall dynamically define rules or establish sessions allowing network traffic to flow for the following network protocols [selection: *FTP, SIP, H.323: [assignment: other supported protocols]*].

Application Note 60

This element requires the specification of more complex protocols that require the firewall to allow network traffic flow even though an existing rule does not explicitly allow the flow. For example, the FTP protocol requires both a control connection and a data connection if a user is to transfer files. While there are well-known ports involved, port 21 (control port on FTP server) and port 20 (data port on server in active mode), there are random ports > 1023 used on the client side. In passive mode, the FTP server may use a random port >1023 instead of port 20. The data connection is initiated by the client in passive mode, and imitated by the FTP server in active mode.

For these types of protocols, the establishment of a "new" connection is allowed, even though the ruleset may appear to deny it (e.g., since a rule cannot predict which random port will be used by the client or potentially the server, the default rule to deny may appear to apply). The TSF could create a dynamic rule that governs the traffic flow, or the TSF could implicitly allow the new connection to be established based on expectations of the protocol implementation as specified in the RFC or equivalent standard.

It is important to note that there is no expectation that any network packets be inspected beyond layer 4 (TCP/UDP). This requirement simply requires that the ST author specify the conditions under which a rule is dynamically inserted into the firewall to allow expected connections with unpredictable UDP/TCP ports to correctly be established.

If the ST Author includes additional protocols they must identify the RFC or equivalent standard that specifies the behavior of the protocol, as is done for FTP above.

B. Selection-Based Requirements

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

B.1 Audit Events for Selection-Based SFRs

Requirement	Auditable Events	Additional Audit Record Contents
FCS_DTLSC_EXT.1	Failure to establish a DTLS session	Reason for failure
FCS_DTLSC_EXT.2	Failure to establish a DTLS session	Reason for failure
FCS_DTLSC_EXT.2	Detected replay attacks	Identity (e.g., source IP address) of the source of the replay attack.
FCS_DTLSS_EXT.1	Failure to establish a DTLS session	Reason for failure
FCS_DTLSS_EXT.1	Detected replay attacks	Identity (e.g., source IP address) of the source of the replay attack.
FCS_DTLSS_EXT.2	Failure to establish a DTLS session	Reason for failure
FCS_DTLSS_EXT.2	Detected replay attacks	Identity (e.g., source IP address) of the source of the replay attack.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure
FCS_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure
FCS_SSHC_EXT.1	Failure to establish an SSH session	Reason for failure

FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSC_EXT.2	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.2	Failure to establish a TLS Session	Reason for failure
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate	Reason for failure
FIA_X509_EXT.2	None	None
FIA_X509_EXT.3	None.	None.
FPT_TST_EXT.2	Failure of self-test	Reason for failure (including identifier of invalid certificate)
FPT_TUD_EXT.2	Failure of update	Reason for failure (including identifier of invalid certificate)
FMT_MOF.1/AutoUpdate	Enabling or Disabling automatic checking for updates or automatic updates.	None.
FMT_MOF.1/Functions	Modification of the behaviour of the transmission of audit data to an external IT entity, the handling of audit data, the audit functionality when Local Audit Storage Space is full.	None.

Table 5: Selection-Based SFRs and Auditable Events

Application Note 61

The audit event for FIA_X509_EXT.1/Rev is based on the TOE not being able to complete the certificate validation by ensuring the following:

- *the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.*
- *Verification of the digital signature of the trusted hierarchical CA*
- *read/access the CRL or access the OCSP server (according to selections in the ST).*

If any of these checks fails, then an audit event with the failure should be written to the audit log.

B.2 Cryptographic Support (FCS)

B.2.1 Cryptographic Protocols (Extended – FCS_DTLSC_EXT, FCS_DTLSS_EXT, FCS_HTTPS_EXT, FCS_ IPSEC_EXT, FCS_SSHC_EXT, FCS_SSHS_EXT, FCS_TLSC_EXT, FCS_TLSS_EXT)

B.2.1.1 FCS_DTLSC_EXT & FCS_DTLSS_EXT DTLS Protocol

Datagram TLS (DTLS) is not a required component of the NDcPP. If a TOE implements DTLS, a corresponding selection in FTP_ITC.1, FTP_TRP.1/Admin, or FPT_ITT.1 should be made to define what the DTLS protocol is implemented to protect.

A TOE may act as the client, the server, or both in DTLS sessions. The requirement has been separated into DTLS Client (FCS_DTLSC_EXT) and DTLS Server (FCS_DTLSS_EXT) requirements to allow for these differences.

If the TOE acts as the client during the claimed DTLS sessions, the ST author should claim one of the FCS_DTLSC_EXT requirements. If the TOE only transmits application-layer data to an external entity using a trusted channel provided by DTLS, (i.e. transmits syslog over DTLS) then FCS_DTLSC_EXT.1 should be selected.

If the application layer communication is bi-directional, that is, the TOE both transmits and receives application data or is managed by the DTLS Server, then FCS_DTLSC_EXT.2 is required. FCS_DTLSC_EXT.2 requires the client must be capable of the following:

- Present a certificate to a DTLS Server for mutual authentication.
- Perform a selected action if a DTLS message from the DTLS Server contains an invalid Message Authentication Code (MAC).
- Detect replayed messages

To ensure audit requirements are properly met, a DTLS receiver may need to monitor the DTLS connection state at the application layer. When no data is received from a DTLS connection for a long time (where the application decides what "long" means), the receiver should send a close_notify alert message and close the connection.

FCS_DTLSC_EXT.1	DTLS Client Protocol

FCS_DTLSC_EXT.1.1 The TSF shall implement [selection: DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)] supporting the following ciphersuites:

- [selection:
  - TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
  - TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

  ].

Application Note 62

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSC_EXT.1.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125 section 6.

Application Note 63

The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 DTLS 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 Evaluation Activity.

FCS_DTLSC_EXT.1.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, *[assignment: other action]*].

Application Note 64

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSC_EXT.1.4 The TSF shall [selection: not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves] in the Client Hello.

Application Note 65

If ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.1.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.1.1, then "not present the Supported Elliptic Curves Extension" should be selected.

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

FCS_DTLSC_EXT.2	**DTLS Client Protocol – with authentication**

FCS_DTLSC_EXT.2.1 The TSF shall implement [selection: DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)] supporting the following ciphersuites:

- [selection:
  - TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
  - TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

].

Application Note 66

The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125 section 6.

Application Note 67

The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 DTLS 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 Evaluation Activity.

FCS_DTLSC_EXT.2.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, *[assignment: other action]*].

Application Note 68

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSC_EXT.2.4 The TSF shall [selection: not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves] in the Client Hello.

Application Note 69

If ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.2.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.2.1, then "not present the Supported Elliptic Curves Extension" should be selected.

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

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

Application Note 70

The use of X.509v3 certificates for DTLS is addressed in FIA_X509_EXT.2.1. This requirement adds that the client must be capable of presenting a certificate to a DTLS server for DTLS mutual authentication.

FCS_DTLSC_EXT.2.6 The TSF shall [selection: terminate the DTLS session, silently discard the record] if a message received contains an invalid MAC.

Application Note 71

The Message Authentication Code (MAC) is keyed hash function specified in FCS_COP.1/KeyedHash. The MAC is negotiated during DTLS handshake phase and is used to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.

FCS_DTLSC_EXT.2.7 The TSF shall detect and silently discard replayed messages for:

- DTLS records previously received.
- DTLS records too old to fit in the sliding window.

Application Note 72

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains a sequence number that is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.

"Silently Discard" means the TOE discards the packet without responding.

FCS_DTLSS_EXT.1	**DTLS Server Protocol**

FCS_DTLSS_EXT.1.1 The TSF shall implement [selection: DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)] supporting the following ciphersuites:

- [selection:
  - TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
  - TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

  ].

Application Note 73

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSS_EXT.1.2 The TSF shall deny connections from clients requesting [none].

Application Note 74

This version of the cPP does not require the TOE to deny DTLS v1.0. In a future version of this cPP DTLS v1.0 will be required to be denied for all TOEs.

FCS_DTLSS_EXT.1.3 The TSF shall not proceed with a connection handshake attempt if the DTLS Client fails validation.

Application Note 75

The process to validate the DTLS client is specified in section 4.2.1 of RFC 6347 (DTLS 1.2) and RFC 4347 (DTLS 1.0). The TOE validates the DTLS client during Connection Establishment (Handshaking) and prior to the TSF sending a Server Hello message. After receiving a ClientHello, the DTLS Server sends a HelloVerifyRequest along with a cookie. The cookie is a signed message using the keyed hash function specified in FCS_COP.1/KeyedHash. The DTLS Client then sends another ClientHello with the cookie attached. If the DTLS server successfully verifies the signed cookie, the Client is not using a spoofed IP address.

FCS_DTLSS_EXT.1.4 The TSF shall [selection: perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size [selection: 2048, bits, 3072 bits]].

Application Note 76

If the ST lists a DHE or ECDHE ciphersuite in FCS_DTLSS_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 to establish the security strength of the DTLS connection.

FCS_DTLSS_EXT.1.5 The TSF shall [selection: terminate the DTLS session, silently discard the record] if a message received contains an invalid MAC.

Application Note 77

The Message Authentication Code (MAC) is keyed hash function specified in FCS_COP.1/KeyedHash. The MAC is negotiated during DTLS handshake phase and is used to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.

FCS_DTLSS_EXT.1.6 The TSF shall detect and silently discard replayed messages for:

- DTLS records previously received.
- DTLS records too old to fit in the sliding window.

Application Note 78

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains *a sequence number that is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.*

"Silently Discard" means the TOE discards the packet without responding.

FCS_DTLSS_EXT.2	DTLS Server Protocol with mutual authentication

FCS_DTLSS_EXT.2.1 The TSF shall implement [selection: DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)] supporting the following ciphersuites:

- [selection:
  - TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
  - TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
  - TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
  - TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
  - TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
  - TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
  - TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
  - TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

  ].

Application Note 79

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSS_EXT.2.2 The TSF shall deny connections from clients requesting [none].

Application Note 80

This version of the cPP does not require the TOE to deny DTLS v1.0. In a future version of this cPP DTLS v1.0 will be required to be denied for all TOEs.

FCS_DTLSS_EXT.2.3 The TSF shall not proceed with a connection handshake attempt if the DTLS Client fails validation.

Application Note 81

The process to validate the DTLS client is specified in section 4.2.1 of RFC 6347 (DTLS 1.2) and RFC 4347 (DTLS 1.0). The TOE validates the DTLS client during Connection Establishment (Handshaking) and prior to the TSF sending a Server Hello message. After receiving a ClientHello, the DTLS Server sends a HelloVerifyRequest along with a cookie. The cookie is a signed message using the keyed hash function specified in FCS_COP.1/KeyedHash. The DTLS Client then sends another ClientHello with the cookie attached. If the DTLS server successfully verifies the signed cookie, the Client is not using a spoofed IP address.

FCS_DTLSS_EXT.2.4 The TSF shall [selection: perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size [selection: 2048, bits, 3072 bits]].

Application Note 82

If the ST lists a DHE or ECDHE ciphersuite in FCS_DTLSS_EXT.2.1, 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 DTLS connection.

FCS_DTLSS_EXT.2.5 The TSF shall [selection: terminate the DTLS session, silently discard the record] if a message received contains an invalid MAC.

Application Note 83

The Message Authentication Code (MAC) is negotiated during the DTLS handshake phase and is used to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.

FCS_DTLSS_EXT.2.6 The TSF shall detect and silently discard replayed messages for:

- DTLS records previously received.
- DTLS records too old to fit in the sliding window.

Application Note 84

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains a sequence number that is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.

"Silently Discard" means the TOE discards the packet without responding.

FCS_DTLSS_EXT.2.7 The TSF shall support mutual authentication of DTLS clients using X.509v3 certificates.

FCS_DTLSS_EXT.2.8 The TSF shall not establish a trusted channel if the client certificate is invalid. If the client certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, *[assignment: other action]*].

Application Note 85

The use of X.509v3 certificates for DTLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include support for client-side certificates for DTLS mutual authentication.

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSS_EXT.2.9 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 client.

Application Note 86

The client identifier may be in the Subject field or the Subject Alternative Name extension of the certificate. The expected identifier may either be configured, may be compared to the Domain Name, IP address, username, or email address used by the peer, or may be passed to a directory server for comparison.

B.2.1.2 FCS_HTTPS_EXT HTTPS Protocol

HTTPS is not a required component of this cPP. If a TOE implements HTTPS, a corresponding selection in FTP_ITC.1, FPT_ITT.1 and/or FTP_TRP.1/Admin should have been made that defines what the HTTPS protocol is implemented to protect.

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 87

The ST author must provide enough detail to determine how the implementation is complying with the standard(s) identified; this can be done by additional detail in the TSS.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 If a peer certificate is presented, the TSF shall [selection: not require client authentication, not establish the connection, request authorization to establish the connection, *[assignment: other action]*]] if the peer certificate is deemed invalid.

Application Note 88

If HTTPS is selected in FTP_TRP.1/Admin or FTP_ITC.1 then 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/Rev. If HTTPS is selected in FPT_ITT.1 then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

B.2.1.3 FCS_IPSEC_EXT.1 IPsec Protocol

The endpoints of network device communication can be geographically and logically distant and may pass through a variety of other potentially untrusted systems. The security functionality of the network device must be able to protect any critical network traffic (administration traffic, authentication traffic, audit traffic, etc.). One way to provide a mutually authenticated communication channel between the network device and an external IT entity is to implement IPsec.

IPsec is not a required component of this cPP. If a TOE implements IPsec, a corresponding selection in FTP_ITC.1, FPT_ITT.1 and/or FTP_TRP.1/Admin should have been made that defines what the IPsec protocol is implemented to protect.

IPsec is a peer to peer protocol and as such does not need to be separated into client and server requirements.

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 89

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 [selection: transport mode, tunnel mode].

Application Note 90

The ST author selects the supported modes of operation for IPsec.

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms [selection: AES-CBC-128, AES-CBC-192, AES-CBC-256 (specified in RFC 3602), no other algorithm] together with a Secure Hash Algorithm (SHA)-based HMAC [selection: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, no other algorithm] and [selection: AES-GCM-128, AES-GCM-192, AES-GCM-256 (specified in RFC 4106), no other algorithm].

Application Note 91

When an AES-CBC algorithm is selected, at least one SHA-based HMAC must also be chosen. If only an AES-GCM algorithm is selected, then a SHA-based HMAC is not required since AES-GCM satisfies both confidentiality and integrity functions. IPsec may utilise a truncated version of the SHA-based HMAC functions contained in the selections. Where a truncated output is utilised, it shall be highlighted in the TSS.

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 92

If the TOE implements SHA-2 hash algorithms for IKEv1 or IKEv2, the ST author selects RFC 4868. If the TOE implements the use of truncated SHA-based HMACs as described in RFC 4868, they shall be highlighted in the TSS.

FCS_IPSEC_EXT.1.6 The TSF shall ensure the encrypted payload in the [selection: IKEv1, IKEv2] protocol uses the cryptographic algorithms [selection: AES-CBC-128, AES-CBC-192, AES-CBC-256 (specified in RFC 3602), AES-GCM-128, AES-GCM-192, AES-GCM-256 (specified in RFC 5282)].

Application Note 93

AES-GCM-128, AES-GCM-192 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 a Security Administrator based on [selection:
    - number of bytes;
    - length of time, where the time values can be configured within [*assignment: integer range including 24*] hours;

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

  ]

].

Application Note 94

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 a Security 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 a Security 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 95

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 96

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 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 97

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 IKE protocols implement DH Group(s) [selection: 14 (2048-bit MODP), 19 (256-bit Random ECP), 20 (384-bit Random ECP), 24 (2048-bit MODP with 256-bit POS)].

Application Note 98

The selection is used to specify additional DH groups supported. This applies to IKEv1 and IKEv2 exchanges.

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 99

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 100

At least one public-key-based Peer Authentication method is required in order to conform to this cPP; 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 if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following types: [selection: IP address, Fully Qualified Domain Name (FQDN), user FQDN, Distinguished Name (DN)] and [selection: no other reference identifier type, [*assignment: other supported reference identifier types*]].

Application Note 101

When using RSA or ECDSA certificates for peer authentication, the reference and presented identifiers take the form of either a DN, IP address, FQDN or user FQDN. The reference identifier is the identifier the TOE expects to receive from the peer during IKE authentication. The presented identifier is the identifier that is contained within the peer certificate body. The ST author shall select the presented and reference identifier types supported and may *optionally assign additional supported identifier types in the second selection. Excluding the DN identifier type (which is necessarily the Subject DN in the peer certificate), the TOE may support the identifier in either the Common Name or Subject Alternative Name (SAN) or both.*

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

Supported peer certificate algorithms are the same as FCS_IPSEC_EXT.1.13

B.2.1.4 FCS_SSHC_EXT & FCS_SSHS_EXT SSH Protocol

SSH is not a required component of this cPP. If a TOE implements SSH, a corresponding selection in FTP_ITC.1, FPT_ITT.1 and/or FTP_TRP.1/Admin should have been made that defines what the SSH protocol is implemented to protect.

A TOE may act as the client or the server in an SSH session. The requirement has been separated into SSH Client (FCS_SSHC_EXT) and SSH Server (FCS_SSHS_EXT) requirements to allow for these differences.

FCS_SSHC_EXT.1	SSH Client Protocol

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

Application Note 102

The ST author selects which of the 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 Evaluation 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, [selection: password-based, no other method].

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 103

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: [selection: aes128-cbc, aes256-cbc, aes128-ctr, aes256-ctr, AEAD_AES_128_GCM, AEAD_AES_256_GCM].

Application Note 104

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. Corresponding FCS_COP entries are included in the ST for the algorithms selected here.

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

Application Note 105

If x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384 or x509v3-ecdsa-sha2-nistp521 are selected, then the list of trusted certification authorities must be selected in FCS_SSHC_EXT.1.9 and the FIA_X509_EXT SFRs in Appendix B are applicable.

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 106

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 within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

Application Note 107

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in *the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.*

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 108

The list of trusted certification authorities can only be selected if x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384 or x509v3-ecdsa-sha2-nistp521 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 RFC(s) [selection: 4251, 4252, 4253, 4254, 5647, 5656, 6187, 6668, no other RFCs].

Application Note 109

The ST author selects which of the 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 Evaluation 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 110

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: [selection: aes123-cbc, aes256-cbc, aes128-ctr, aes256-ctr, AEAD_AES_128_GCM, AEAD_AES_256_GCM].

Application Note 111

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. Corresponding FCS_COP entries are included in the ST for the algorithms selected here.

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [selection: ssh-rsa, ecdsa-sha2-nistp256] and [selection: ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521, no other public key algorithms] as its public key algorithm(s) and rejects all other public key algorithms.

Application Note 112

If x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384 or x509v3-ecdsa-sha2-nistp521 are selected then the FIA_X509_EXT SFRs in Appendix B are applicable

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 113

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 within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

Application Note 114

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.

B.2.1.5 FCS_TLSC_EXT & FCS_TLSS_EXT TLS Protocol

TLS is not a required component of this cPP. If a TOE implements TLS, a corresponding selection in FPT_ITT.1, FTP_ITC.1, or FTP_TRP.1/Admin should be made to define what the TLS protocol is implemented to protect.

A TOE may act as the client, the server, or both in TLS sessions. The requirement has been separated into TLS Client (FCS_TLSC_EXT) and TLS Server (FCS_TLSS_EXT) requirements to allow for these differences. If the TOE acts as the client during the claimed TLS sessions, the ST author should claim one of the FCS_TLSC_EXT requirements. If the TOE acts as the server during the claimed TLS sessions, the ST author should claim one of the FCS_TLSS_EXT requirements. If the TOE acts as both a client and server during the claimed TLS sessions, the ST author should claim one of the FCS_TLSC_EXT and FCS_TLSS_EXT requirements.

Additionally, TLS may or may not be performed with client authentication. The ST author shall claim FCS_TLSC_EXT.1 and/or FCS_TLSS_EXT.1 if the TOE does not support client authentication. The ST author should claim FCS_TLSC_EXT.2 and/or FCS_TLSS_EXT.2 if client authentication is performed by the TOE.

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)] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246

- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

].

Application Note 115

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

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 per RFC 6125 section 6.

Application Note 116

The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 Evaluation Activity.

FCS_TLSC_EXT.1.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, [*assignment: other action*]].

Application Note 117

If TLS is selected in FTP_TRP.1/Admin or FTP_ITC then 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/Rev. If TLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT

FCS_TLSC_EXT.1.4 The TSF shall [selection: not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves] in the Client Hello.

Application Note 118

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/SigGen and FCS_CKM.1 and FCS_CKM.2. This extension is required for clients supporting Elliptic Curve ciphersuites.

FCS_TLSC_EXT.2	TLS Client Protocol with authentication

FCS_TLSC_EXT.2.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[selection:
- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246

- TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

].

Application Note 119

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

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

FCS_TLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier per RFC 6125 section 6.

Application Note 120

The rules for verification of identify are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 Evaluation Activity.

FCS_TLSC_EXT.2.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, *[assignment: other action]*].

Application Note 121

If TLS is selected in FTP_TRP.1/Admin or FTP_ITC then validity is determined by the identifier verification, certificate path, the expiration date, and the revocation status in accordance with RFC 5280. Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1/Rev. If TLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT

FCS_TLSC_EXT.2.4 The TSF shall [selection: not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves] in the Client Hello.

Application Note 122

If ciphersuites with elliptic curves were selected in FCS_TLSC_EXT.2.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_TLSC_EXT.2.1, 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/SigGen and FCS_CKM.1 and FCS_CKM.2. This extension is required for clients supporting Elliptic Curve ciphersuites.

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

Application Note 123

The use of X.509v3 certificates for TLS is addressed in FIA_X509_EXT.2.1. This requirement adds that the client must be capable of presenting a certificate to a TLS server for TLS mutual authentication.

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)] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[selection:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

].

Application Note 124

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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

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 2.0, SSL 3.0, TLS 1.0 and [selection: TLS 1.1, TLS 1.2, none].

Application Note 125

All SSL versions and TLS v1.0 are denied. Any TLS versions not selected in FCS_TLSS_EXT.1.1 should be selected here. (If "none" is the selection for this element then the ST author may omit the words "and none".)

FCS_TLSS_EXT.1.3 The TSF shall [selection: perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size [selection: 2048, bits, 3072 bits]].

Application Note 126

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 to establish the security strength of the TLS connection.

FCS_TLSS_EXT.2	**TLS Server Protocol with mutual authentication**

FCS_TLSS_EXT.2.1 The TSF shall implement [selection: TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[selection:
- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_192_CBC_SHA as defined in RFC 4492
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_192_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_192_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288

- TLS_RSA_WITH_AES_192_GCM_SHA256 as defined in RFC 5288
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_192_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_192_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_192_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_192_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

].

Application Note 127

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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 5246.

These requirements will be revisited as new TLS versions are standardized by the IETF.

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

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

Application Note 128

All SSL versions and TLS v1.0 are denied. Any TLS versions not selected in FCS_TLSS_EXT.1.1 should be selected here. (If "none" is the selection for this element then the ST author may omit the words "and none".)

FCS_TLSS_EXT.2.3 The TSF shall [selection: perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size 2048 bits and [selection: 3072 bits, no other size]].

Application Note 129

If the ST lists a DHE or ECDHE ciphersuite in FCS_TLSS_EXT.2.1, 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 to establish the security strength of the TLS connection.

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 client certificate is invalid. If the client certificate is deemed invalid, then the TSF shall [selection: not establish the connection, request authorization to establish the connection, *[assignment: other action]*].

Application Note 130

The use of X.509v3 certificates for TLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include support for client-side certificates for TLS mutual authentication.

If TLS is selected for FTP_TRP or FTP_ITC then validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280. Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1/Rev. If TLS is selected for FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT

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 client.

Application Note 131

The client identifier may be in the Subject field or the Subject Alternative Name extension of the certificate. The expected identifier may either be configured, may be compared to the Domain Name, IP address, username, or email address used by the client, or may be passed to a directory server for comparison.

B.3 Identification and Authentication (FIA)

B.3.1 Authentication using X.509 certificates (Extended – FIA_X509_EXT)

X.509 certificate-based authentication is required if IPsec or TLS communications are claimed for FPT_ITT, FTP_ITC.1 or FTP_TRP. These SFRs are also required if FPT_TUD_EXT.2 or FPT_TST_EXT.2 are claimed. If SSH client communications are claimed and any x509 algorithms are claimed in FCS_SSHC_EXT.1.5 or FCS_SSHS_EXT.1.5, these SFRs are required. In the case of the TOE only acting as the SSH server or acting as the client, but not claiming any x509 algorithms in FCS_SSHC_EXT.1.5 or FCS_SSHS_EXT.1.5, these SFRs are optional.

B.3.1.1 FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1/Rev	**X.509 Certificate Validation**

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

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of three certificates.
- 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 6960, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5]

- 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/Rev 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.

Application Note 132

FIA_X509_EXT.1.1/Rev lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. The trusted channel/path protocols may require that certificates are used; this use requires that the extendedKeyUsage rules are verified. If the TOE does not support functionality that uses any of the certificate types listed in the extendedKeyUsage rules in FIA_X509_EXT.1.1 then this is stated in the TSS and the relevant part of the SFR is considered trivially satisfied. However, if the TOE does support functionality that uses certificates of any of these types then the corresponding rule must be satisfied as in the SFR.

The TOE shall be capable of supporting a minimum path length of three certificates. That is, it shall support a hierarchy comprising of at least a self-signed root certificate, a subordinate CA certificate and a TOE identity certificate.

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

The TSS shall describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the device.

It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

FIA_X509_EXT.1.2/Rev applies to certificates that are used and processed by the TSF and restricts the certificates that may be added as trusted CA certificates.

The ST author must include FIA_X509_EXT.1/Rev in all instances except when only SSH is selected within FTP_ITC.1 or FPT_ITT.1 and authentication is limited to ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and/or ecdsa-sha2-nistp521 Additionally, FIA_X509_EXT.1/Rev must also be included if either FPT_TUD_EXT or FPT_TST_EXT have selected to use X509 certificates.

B.3.1.2 FIA_X509_EXT.2 X.509 Certificate Authentication

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: DTLS, HTTPS, IPsec, SSH, TLS], and [selection: code signing for system software updates, code signing for integrity verification, *[assignment: other uses],* 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].

Application Note 133

In FIA_X509_EXT.2.1, the ST author's selection includes IPsec, TLS, or HTTPS if these protocols are included in FTP_ITC.1.1 or FPT_ITT.1. SSH should be included if authentication other than ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and/or ecdsa-sha2-nistp521 is selected in FCS_SSHC_EXT.1.5 or FCS_SSHS_EXT.1.5. Certificates may optionally be used for trusted updates of system software (FPT_TUD_EXT.2) and for integrity verification (FPT_TST_EXT.2).

Often a connection must be established to check the revocation status of a certificate - either to download a CRL or to perform a lookup using OCSP. In FIA_X509_EXT.2.2 the selection is used to describe the behaviour in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate valid according to all other rules in FIA_X509_EXT.1, the behaviour indicated in the selection determines the validity. The TOE must not accept the certificate if it fails any of the other validation rules in FIA_X509_EXT.1. If the Administrator-configured option is selected by the ST Author, the ST Author also selects the corresponding function in FMT_SMF.1. The selection should be consistent with the validation requirements in FCS_IPSEC_EXT.1.14, FCS_TLSC_EXT.1.3 and FCS_TLSC_EXT.2.3.

If the TOE is distributed and FIA_X509_EXT.1/ITT is selected, then certificate revocation checking is optional. This is due to additional authorization actions being performed in the enabling and disabling of the intra-TOE trusted channel as defined in FCO_CPC_EXT.1. In this case, a connection is not required to determine certificate validity and this SFR is trivially satisfied.

The ST author must include FIA_X509_EXT.2 in all instances except when only SSH is selected within FTP_ITC.1 or FPT_ITT.1 and ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and/or ecdsa-sha2-nistp521 authentication is also selected. Additionally, FIA_X509_EXT.2 *must also be included if either FPT_TUD_EXT or FPT_TST_EXT have selected X509 certificates.*

B.3.1.3 FIA_X509_EXT.3 X.509 Certificate Requests

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].

Application Note 134

The public key is the public key portion of the public-private key pair generated by the TOE as specified in FCS_CKM.1.

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

B.4 Protection of the TSF (FPT)

B.4.1 TSF self-test (Extended)

B.4.1.1 FPT_TST_EXT.2 Self-tests based on certificates

FPT_TST_EXT.2	**Self-tests based on certificates**

FPT_TST_EXT.2.1 The TSF shall fail self-testing if a certificate is used for self-tests and the corresponding certificate is deemed invalid.

Application Note 135

Certificates may optionally be used for self-tests (FPT_TST_EXT.1.1). This element must be included in the ST if certificates are used for self-tests. If "code signing for integrity verification" is selected in FIA_X509_EXT.2.1, FPT_TST_EXT.2 must be included in the ST.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with FIA_X509_EXT.1/Rev.

B.4.2 Trusted Update (FPT_TUD_EXT)

B.4.2.1 FPT_TUD_EXT.2 Trusted Update based on certificates

FPT_TUD_EXT.2	**Trusted Update based on certificates**

FPT_TUD_EXT.2.1 The TSF shall not install an update if the code signing certificate is deemed invalid.

FPT_TUD_EXT.2.2 When the certificate is deemed invalid because the certificate has expired, 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 136

Certificates may optionally be used for code signing of system software updates (FPT_TUD_EXT.1.3). This element must be included in the ST if certificates are used for validating updates. If "code signing for system software updates" is selected in FIA_X509_EXT.2.1, FPT_TUD_EXT.2 must be included in the ST. The use of X.509 certificates is not applicable if only published hashes are supported for trusted updates.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with FIA_X509_EXT.1/Rev. For expired certificates the author of the ST selects whether the certificate shall be accepted, rejected or the choice is left to the Administrator to accept or reject the certificate.

B.5 Security Management (FMT)

B.5.1 Management of functions in TSF (FMT_MOF)

B.5.1.1 FMT_MOF.1/AutoUpdate Management of security functions behaviour

FMT_MOF.1/AutoUpdate	**Management of security functions behaviour**

FMT_MOF.1.1/AutoUpdate The TSF shall restrict the ability to [selection: enable, disable] the functions [selection: automatic checking for updates, automatic update] to Security Administrators.

Application Note 137

FMT_MOF.1/AutoUpdate is only applicable if the TOE supports automatic checking for updates and/or automatic updates and allows them to be enabled and disabled. Enable and disable of automatic checking for updates and/or automatic updates is restricted to Security Administrators. The option "automatic update" may only be selected if digital signatures are used to validate the trusted update.

B.5.1.2 FMT_MOF.1/Functions Management of security functions behaviour

FMT_MOF.1/Functions Management of security functions behaviour

FMT_MOF.1.1/Functions The TSF shall restrict the ability to [selection: determine the behaviour of, modify the behaviour of] the functions [selection: transmission of audit data to an external IT entity, handling of audit data, audit functionality when Local Audit Storage Space is full] to Security Administrators.

Application Note 138

FMT_MOF.1/Functions should be chosen if one or more of the following scenarios apply:

- *If the transmission protocol for transmission of audit data to an external IT entity as defined in FAU_STG_EXT.1.1 is configurable, "transmission of audit data to an external IT entity" shall be chosen.*
- *If the handling of audit data is configurable, "handling of audit data" shall be chosen. The term "handling of audit data" refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2/LocSpace.*
- *If the behaviour of the audit functionality is configurable when Local Audit Storage Space is full, "audit functionality when Local Audit Storage Space is full" shall be chosen.*

The first selection for "determine the behaviour of" and "modify the behaviour of" should be done as appropriate. It might be necessary to have different selections for the first selection depending on the second selection (e.g. "handling of audit data" might require "determine the behaviour of" and "modify the behaviour of" for the first selection on the one hand and "TOE Security Functions" might require "modify the behaviour of" only). In that case FMT_MOF.1/Functions should be iterated with increasing number appended (i.e. FMT_MOF.1/Functions1, FMT_MOF.1/Functions2, etc.).

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: formatting conventions for selections and assignments in this Appendix are those in [CC2].)

C.1 Security Audit (FAU)

C.1.1 Protected audit event storage (FAU_STG_EXT)

Family Behaviour

This component defines the requirements for the TSF to be able to securely transmit audit data between the TOE and an external IT entity.

Component levelling

FAU_STG_EXT.1 Protected audit event storage requires the TSF to use a trusted channel implementing a secure protocol.

FAU_STG_EXT.2 Counting lost audit data requires the TSF to provide information about audit records affected when the audit log becomes full.

Management: FAU_STG_EXT.1, FAU_STG_EXT.2

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

a) The TSF shall have the ability to configure the cryptographic functionality.

Audit: FAU_STG_EXT.1, FAU_STG_EXT.2

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

a) No audit necessary.

C.1.1.1 FAU_ STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1	Protected Audit Event Storage

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation
FTP_ITC.1 Inter-TSF Trusted Channel

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.

Application Note 139

For selecting the option of transmission of generated audit data to an external IT entity the TOE relies on a non-TOE audit server for storage and review of audit records. The storage of these audit records and the ability to allow the Administrator to review these audit records is provided by the operational environment in that case. Since the external audit server is not part of the TOE, there are no requirements on it except the capabilities for ITC transport for audit data. No requirements are placed upon the format or underlying protocol of the audit data being transferred. The TOE must be capable of being configured to transfer audit data to an external IT entity without Administrator intervention. Manual transfer would not meet the requirements. Transmission could be done in real-time or periodically. If the transmission is not done in real-time then the TSS describes what event stimulates the transmission to be made and what range of frequencies the TOE supports for making transfers of audit data to the audit server; the TSS also suggests typical acceptable frequencies for the transfer.

For distributed TOEs each component must be able to export audit data across a protected channel external (FTP_ITC.1) or intercomponent (FPT_ITT.1 or FTP_ITC.1) as appropriate. At least one component of the TOE must be able to export audit records via FTP_ITC.1 such that all TOE audit records can be exported to an external IT entity.

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself.

FAU_STG_EXT.1.3 The TSF shall [selection: *drop new audit data, overwrite previous audit records according to the following rule: [assignment: rule for overwriting previous audit records], [assignment: other action]*] when the local storage space for audit data is full.

Application Note 140

The external log server might be used as alternative storage space in case the local storage space is full. The "other action" could in this case be defined as "send the new audit data to an external IT entity".

For distributed TOEs each component must provide some amount of local storage to ensure that audit records are preserved in case of network connectivity issues. The behaviour when local storage is exhausted must be described for each component.

FCS_RBG_EXT Random Bit Generation 1

C.1.1.2 FAU_ STG_EXT.2 Counting lost audit data

FAU_STG_EXT.2	**Counting lost audit data**

Hierarchical to:	No other components.
Dependencies:	FAU_GEN.1 Audit data generation FAU_STG_EXT.1 External Audit Trail Storage

FAU_STG_EXT.2.1 The TSF shall provide information about the number of [selection: *dropped, overwritten, assignment: other information*] audit records in the case where the local storage has been filled and the TSF takes one of the actions defined in FAU_STG_EXT.1.3.

Application Note 141

This option should be chosen if the TOE supports this functionality.

In case the local storage for audit records is cleared by the Administrator, the counters associated with the selection in the SFR should be reset to their initial value (most likely to 0). The guidance documentation should contain a warning for the Administrator about the loss of audit data when he clears the local storage for audit records.

For distributed TOEs each component that implements counting of lost audit data has to provide a mechanism for Administrator access to, and management of, this information.

If FAU_STG_EXT.2 is added to the ST, the ST has to make clear any situations in which lost audit data is not counted.

C.2 Cryptographic Support (FCS)

C.2.1 Random Bit Generation (FCS_RBG_EXT)

C.2.1.1FCS_RBG_EXT.1 Random Bit Generation

Family Behaviour

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

Component levelling

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

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

a) There are no management activities foreseen

Audit: FCS_RBG_EXT.1

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

a) Minimal: failure of the randomization process

FCS_RBG_EXT.1	**Random Bit Generation**

Hierarchical to: No other components

Dependencies: No other components

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [selection*: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [selection: *[assignment: number of software-based sources] software-based noise source, [assignment: number of hardware-based sources] hardware-based noise source*] with 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 142

For the first selection in FCS_RBG_EXT.1.2, 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 Evaluation Activity for this element necessarily describes 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-1, 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.

C.2.2 Cryptographic Protocols (FCS_DTLSC_EXT, FCS_DTLSS_EXT, FCS_HTTPS_EXT, FCS_IPSEC_EXT, FCS_SSHC_EXT, FCS_SSHS_EXT, FCS_TLSC_EXT, FCS_TLSS_EXT)

C.2.2.1FCS_DTLSC_EXT DTLS Client Protocol

Family Behaviour

The component in this family addresses the ability for a client to use DTLS to protect data between the client and a server using the DTLS protocol.

Component levelling

FCS_DTLSC_EXT.1 DTLS Client requires that the client side of DTLS be implemented as specified.

FCS_DTLSC_EXT.2 DTLS Client requires that the client side of the DTLS implementation include mutual authentication.

Management: FCS_DTLSC_EXT.1, FCS_DTLSC_EXT.2

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

a) There are no management activities foreseen.

Audit: FCS_DTLSC_EXT.1, FCS_DTLSC_EXT.2

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

a) Failure of DTLS session establishment
b) DTLS session establishment
c) DTLS session termination

FCS_DTLSC_EXT.1	**DTLS Client Protocol**

Hierarchical to: No other components

Dependencies:
FCS_CKM. 1DataEncryption1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen1SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_DTLSC_EXT.1.1 The TSF shall implement [selection: *DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)*] supporting the following ciphersuites:

- [assignment: *List of optional ciphersuites and reference to RFC in which each is defined*]

Application Note 143

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSC_EXT.1.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125 section 6.

Application Note 144

The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 DTLS server's certificate.

FCS_DTLSC_EXT.1.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 145

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSC_EXT.1.4 The TSF shall [selection: *not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves*] in the Client Hello.

Application Note 146

If ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.1.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_DTLS_EXT.1.1, then "not present the Supported Elliptic Curves Extension" should be selected.

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

FCS_DTLSC_EXT.2	**DTLS Client Protocol with Authentication**

Hierarchical to: FCS_DTLSC_EXT.1 DTLS Client Protocol

Dependencies: FCS_CKM.1/DataEncryption Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_DTLSC_EXT.2.1 The TSF shall implement [selection: *DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)*] supporting the following ciphersuites:

- [assignment: *List of optional ciphersuites and reference to RFC in which each is defined*].

Application Note 147

The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125 section 6.

Application Note 148

The rules for verification of identity are described in Section 6 of RFC 6125. The reference identifier is established by the Administrator (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 DTLS server's certificate.

FCS_DTLSC_EXT.2.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 149

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSC_EXT.2.4 The TSF shall [selection: *not present the Supported Elliptic Curves Extension, present the Supported Elliptic Curves Extension with the following NIST curves: [selection: secp256r1, secp384r1, secp521r1] and no other curves*] in the Client Hello].

Application Note 150

If ciphersuites with elliptic curves were selected in FCS_DTLSC_EXT.2.1, a selection of one or more curves is required. If no ciphersuites with elliptic curves were selected in FCS_DTLS_EXT.2.1, then "not present the Supported Elliptic Curves Extension" should be selected.

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

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

Application Note 151

The use of X.509v3 certificates for TLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include the client must be capable of presenting a certificate to a DTLS server for DTLS mutual authentication.

FCS_DTLSC_EXT.2.6 The TSF shall [selection: *terminate the DTLS session, silently discard the record*] if a message received contains an invalid MAC.

Application Note 152

The Message Authentication Code (MAC) is keyed hash function specified in FCS_COP.1/KeyedHash. The MAC is negotiated during DTLS handshake phase and is used to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.

FCS_DTLSC_EXT.2.7 The TSF shall detect and silently discard replayed messages for:

- DTLS records previously received.
- DTLS records too old to fit in the sliding window.

Application Note 153

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains a sequence number is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.

"Silently Discard" means the TOE discards the packet responding.

C.2.2.2 FCS_DTLSS_EXT DTLS Server Protocol

Family Behaviour

The component in this family addresses the ability for a server to use DTLS to protect data between a client and the server using the DTLS protocol.

Component levelling

FCS_DTLSS_EXT.1 DTLS Server requires that the server side of TLS be implemented as specified.

FCS_DTLSS_EXT.2: DTLS Server requires the mutual authentication be included in the DTLS implementation.

Management: FCS_DTLSS_EXT.1, FCS_DTLSS_EXT.2

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

a) There are no management activities foreseen.

Audit: FCS_DTLSS_EXT.1, FCS_DTLSS_EXT.2

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

a) Failure of DTLS session establishment.
b) DTLS session establishment
c) DTLS session termination

FCS_DTLSS_EXT.1	**DTLS Server Protocol**

Hierarchical to: No other components

Dependencies:
FCS_CKM.1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1//DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1//SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_DTLSS_EXT.1.1 The TSF shall implement [selection: *DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)*] supporting the following ciphersuites:

- [assignment: *List of optional ciphersuites and reference to RFC in which each is defined*]

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. *TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.*

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSS_EXT.1.2 The TSF shall deny connections from clients requesting [assignment: *list of protocol versions*].

Application Note 154

This version of the cPP does not require the TOE to deny DTLS v1.0. In a future version of this cPP DTLS v1.0 will be required to be denied for all TOEs.

FCS_DTLSS_EXT.1.3 The TSF shall not proceed with a connection handshake attempt if the DTLS Client fails validation.

Application Note 155

The process to validate the IP address of a DTLS client is specified in section 4.2.1 of RFC 6347 (DTLS 1.2) and RFC 4347 (DTLS 1.0). The TOE validates the DTLS client during Connection Establishment (Handshaking) and prior to the TSF sending a Server Hello message. After receiving a ClientHello, the DTLS Server sends a HelloVerifyRequest along with a cookie. The cookie is a signed message using the keyed hash function specified in FCS_COP.1 /KeyedHash. The DTLS Client then sends another ClientHello with the cookie attached. If the DTLS server successfully verifies the signed cookie, the Client is not using a spoofed IP address.

FCS_DTLSS_EXT.1.4 The TSF shall [selection: *perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size [selection: 2048, bits, 3072 bits]*].

Application Note 156

If the ST lists a DHE or ECDHE ciphersuite in FCS_DTLSS_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 DTLS connection.

FCS_DTLSS_EXT.1.5 The TSF shall [selection: *terminate the DTLS session, silently discard the record*] if a message received contains an invalid MAC.

Application Note 157

The Message Authentication Code (MAC) is keyed hash function specified in FCS_COP.1/KeyedHash. The MAC is negotiated during DTLS handshake phase and is used *to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.*

FCS_DTLSS_EXT.1.6 The TSF shall detect and silently discard replayed messages for:

- DTLS records previously received.
- DTLS records too old to fit in the sliding window.

Application Note 158

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains a sequence number is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.

"Silently Discard" means the TOE discards the packet without responding.

FCS_DTLSS_EXT.2	**DTLS Server Protocol with mutual authentication**

Hierarchical to: FCS_DTLSS_EXT.1 DTLS Server Protocol

Dependencies:
FCS_CKM.1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1//DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1//SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_DTLSS_EXT.2.1 The TSF shall implement [selection: *DTLS 1.2 (RFC 6347), DTLS 1.0 (RFC 4347)*] supporting the following ciphersuites:

- [assignment: *List of optional ciphersuites and reference to RFC in which each is defined*].

Application Note 159

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the 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. TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0 compliance; however, it is required if claiming compliance with RFC 6347.

These requirements will be revisited as new DTLS versions are standardized by the IETF.

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

FCS_DTLSS_EXT.2.2 The TSF shall deny connections from clients requesting [assignment: *list of protocol versions*].

Application Note 160

This version of the cPP does not require the TOE to deny DTLS v1.0. In a future version of this cPP DTLS v1.0 will be required to be denied for all TOEs.

FCS_DTLSS_EXT.2.3 The TSF shall not proceed with a connection handshake attempt if the DTLS Client fails validation.

Application Note 161

The process to validate the IP address of a DTLS client is specified in section 4.2.1 of RFC 6347 (DTLS 1.2) and RFC 4347 (DTLS 1.0). The TOE validates the DTLS client during Connection Establishment (Handshaking) and prior to the TSF sending a Server Hello message. After receiving a ClientHello, the DTLS Server sends a HelloVerifyRequest along with a cookie. The cookie is a signed message using the keyed hash function specified in FCS_COP.1/KeyedHash. The DTLS Client then sends another ClientHello with the cookie attached. If the DTLS server successfully verifies the signed cookie, the Client is not using a spoofed IP address.

FCS_DTLSS_EXT.2.4 The TSF shall [selection: *perform RSA key establishment with key size [selection: 2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [selection: secp256r1, secp384r1, secp521r1] and no other curves; generate Diffie-Hellman parameters of size [selection: 2048, bits, 3072 bits]*].

Application Note 162

If the ST lists a DHE or ECDHE ciphersuite in FCS_DTLSS_EXT.2.1, 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 DTLS connection.

FCS_DTLSS_EXT.2.5 The TSF shall [selection: *terminate the DTLS session, silently discard the record*] if a message received contains an invalid MAC.

Application Note 163

The Message Authentication Code (MAC) is negotiated during the DTLS handshake phase and is used to protect integrity of messages received from the sender during DTLS data exchange. If MAC verification fails, the session must be terminated or the record must be silently discarded.

FCS_DTLSS_EXT.2.6 The TSF shall detect and silently discard replayed messages for:

- DTLS records that have previously been received.
- DTLS records too old to fit in the sliding window.

Application Note 164

Replay Detection is described in section 4.1.2.6 of DTLS 1.2 (RFC 6347) and section 4.1.2.5 of DTLS 1.0 (RFC 4347). For each received record, the receiver verifies the record contains a sequence number is within the sliding receive window and does not duplicate the sequence number of any other record received during the session.

"Silently Discard" means the TOE discards the packet without responding.

FCS_HTTPS_EXT HTTPS Protocol 1

FCS_DTLSS_EXT.2.7 The TSF shall support mutual authentication of DTLS clients using X.509v3 certificates.

FCS_DTLSS_EXT.2.8 The TSF shall not establish a trusted channel if the client certificate is invalid. If the client certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 165

The use of X.509v3 certificates for DTLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include support for client-side certificates for DTLS mutual authentication.

If DTLS is selected in FTP_ITC then 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/Rev. If DTLS is selected in FPT_ITT, then certificate validity is tested in accordance with testing performed for FIA_X509_EXT.1/ITT.

FCS_DTLSS_EXT.2.9 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 client.

Application Note 166

The client identifier may be in the Subject field or the Subject Alternative Name extension of the certificate. The expected identifier may either be configured, may be compared to the Domain Name, IP address, username, or email address used by the peer, or may be passed to a directory server for comparison. Matching should be performed by a bit-wise comparison.

C.2.2.3 FCS_HTTPS_EXT.1 HTTPS Protocol

Family Behaviour

Components in this family define the requirements for protecting remote management sessions between the TOE and a Security Administrator. This family describes how HTTPS will be implemented. This is a new family defined for the FCS Class.

Component levelling

FCS_HTTPS_EXT.1 HTTPS requires that HTTPS be implemented according to RFC 2818 and supports TLS.

Management: FCS_HTTPS_EXT.1

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

a) There are no management activities foreseen.

FCS_IPSEC_EXT IPsec Protocol 1

Audit: FCS_HTTPS_EXT.1

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

a) There are no auditable events foreseen.

FCS_HTTPS_EXT.1	**HTTPS Protocol**

Hierarchical to: No other components

Dependencies: [FCS_TLSC_EXT.1 TLS Client Protocol, or
FCS_TLSS_EXT.1 TLS Server Protocol]

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 the HTTPS protocol using TLS.

FCS_HTTPS_EXT.1.3 The TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*] if the peer certificate is deemed invalid.

C.2.2.4 FCS_IPSEC_EXT.1 IPsec Protocol

Family Behaviour

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

Component levelling

FCS_IPSEC_EXT IPsec Protocol — 1

FCS_IPSEC_EXT.1 IPsec requires that IPsec be implemented as specified.

Management: FCS_IPSEC_EXT.1

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

a) Maintenance of SA lifetime configuration

Audit: FCS_IPSEC_EXT.1

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

a) Decisions to DISCARD, BYPASS, PROTECT network packets processed by the TOE.
b) Failure to establish an IPsec SA
c) IPsec SA establishment

d) IPsec SA termination
e) Negotiation "down" from an IKEv2 to IKEv1 exchange.

FCS_IPSEC_EXT.1	**Internet Protocol Security (IPsec) Communications**

Hierarchical to: No other components

Dependencies: FCS_CKM.1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

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

Application Note 167

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 [selection: *tunnel mode, transport mode*].

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms [selection: *AES-CBC-128, AES-CBC-192, AES-CBC-256 (specified in RFC 3602), no other algorithm*] together with a Secure Hash Algorithm (SHA)-based HMAC [selection: *HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, no other algorithm*] and [selection: *AES-GCM-128, AES-GCM-192, AES-GCM-256 (specified in RFC 4106), no other algorithm*].

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 RFCs 5996 [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 [selection: *AES-CBC-128, AES_CBC-192 AES-CBC-256 (specified in RFC 3602), AES-GCM-128, AES-GCM-192, AES-GCM-256 (specified in RFC 5282)*].

Application Note 168

AES-GCM-128, AES-GCM-192 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 a Security Administrator based on [selection:*
    - *number of bytes;*
    - *length of time, where the time values can be configured within [assignment: integer range including 24] hours;*

  ];

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

  ]

].

Application Note 169

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 a Security 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 a Security 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 170

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 171

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 172

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 IKE protocols implement DH Group(s) [selection: 14 (2048-bit MODP), 19 (256-bit Random ECP), 24 (2048-bit MODP with 256-bit POS), 20 (384-bit Random ECP)] and [selection: 5 (1536-bit MODP), no other group].

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 173

The ST author chooses either or both of the IKE selections based on what is implemented by the TOE. 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*].

FCS_IPSEC_EXT.1.14 The TSF shall only establish a trusted channel if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following types: [selection: *IP address, Fully Qualified Domain Name (FQDN), user FQDN, Distinguished Name (DN)*] and [selection: *no other reference identifier type, [assignment: other supported reference identifier types]*].

C.2.2.5 FCS_SSHC_EXT.1 SSH Client

Family Behaviour

The component in this family addresses the ability for a client to use SSH to protect data between the client and a server using the SSH protocol.

Component levelling

FCS_SSHC_EXT SSH Client Protocol — 1

FCS_SSHC_EXT.1 SSH Client requires that the client side of SSH be implemented as specified.

Management: FCS_SSHC_EXT.1

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

a) There are no management activities foreseen.

Audit: FCS_SSHC_EXT.1

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

a) Failure of SSH session establishment
b) SSH session establishment
c) SSH session termination

FCS_SSHC_EXT.1	**SSH Client Protocol**

Hierarchical to: No other components

Dependencies:
FCS_CKM.1Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

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

Application Note 174

The ST author selects which of the 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 Evaluation 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, [selection: *password-based, no other method*].

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 175

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: [assignment: *list of encryption algorithms*].

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [selection: *ssh-rsa, ecdsa-sha2-nistp256*] and [selection: *ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521, 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 [assignment: *list of data integrity 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 [assignment: *list of key exchange methods*] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHC_EXT.1.8 The TSF shall ensure that within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

Application Note 176

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.

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

FCS_SSHS_EXT SSH Server Protocol 1

[selection: *public key, a list of trusted certification authorities, no other methods*] as described in RFC 4251 section 4.1.

Application Note 177

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

C.2.2.6 FCS_SSHS_EXT.1 SSH Server Protocol

Family Behaviour

The component in this family addresses the ability for a server to offer SSH to protect data between a client and the server using the SSH protocol.

Component levelling

FCS_SSHS_EXT SSH Server Protocol — 1

FCS_SSHS_EXT.1 SSH Server requires that the server side of SSH be implemented as specified.

Management: FCS_SSHS_EXT.1

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

a) There are no management activities foreseen.

Audit: FCS_SSHS_EXT.1

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

a) Failure of SSH session establishment
b) SSH session establishment
c) SSH session termination

FCS_SSHS_EXT.1	**SSH Server Protocol**

Hierarchical to: No other components

Dependencies: FCS_CKM.1Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)

FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

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

Application Note 178

The ST author selects which of the 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 Evaluation 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 179

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: [assignment: *encryption algorithms*].

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [selection: *ssh-rsa, ecdsa-sha2-nistp256*] and [selection: *ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521, 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 [assignment: *list of MAC algorithms*] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7 The TSF shall ensure that [assignment: *list of key exchange methods*] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8 The TSF shall ensure that within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

FCS_TLSC_EXT TLS Client Protocol 1 2

Application Note 180

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.

C.2.2.7 FCS_TLSC_EXT TLS Client Protocol

Family Behaviour

The component in this family addresses the ability for a client to use TLS to protect data between the client and a server using the TLS protocol.

Component levelling

FCS_TLSC_EXT.1 TLS Client requires that the client side of TLS be implemented as specified.

FCS_TLSC_EXT.2 TLS Client requires that the client side of the TLS implementation include mutual authentication.

Management: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2

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

a) There are no management activities foreseen.

Audit: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2

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

a) Failure of TLS session establishment
b) TLS session establishment
c) TLS session termination

FCS_TLSC_EXT.1	TLS Client Protocol

Hierarchical to: No other components

Dependencies: FCS_CKM. 1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_TLSC_EXT.1.1 The TSF shall implement [selection: *TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TSF shall support the following ciphersuites: [selection:

- *Mandatory Ciphersuites:*
  - *[assignment: list of mandatory ciphersuites and reference to RFC in which each is defined]*
- *[selection: Optional Ciphersuites:*
  - *[assignment: list of optional ciphersuites and reference to RFC in which each is defined];*

  no other ciphersuites]]].

Application Note 181

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. Note that TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0, but is required to ensure compliance with RFC 5246.

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

Application Note 182

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.

FCS_TLSC_EXT.1.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 183

Validity is determined by the identifier verification, certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

FCS_TLSC_EXT.1.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [assignment: *list of supported curves including an option for "none"*].

Application Note 184

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_TLS_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/SigGen and FCS_CKM.1 and FCS_CKM.2. This extension is required for clients supporting Elliptic Curve ciphersuites.

FCS_TLSC_EXT.2	**TLS Client Protocol with Authentication**

Hierarchical to: FCS_TLSC_EXT.1 TLS Client Protocol

Dependencies:
FCS_CKM.1Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_TLSC_EXT.2.1 The TSF shall implement [selection: *TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TSF shall support the following ciphersuites: [selection:

- *Mandatory Ciphersuites:*
  - *[assignment: list of mandatory ciphersuites and reference to RFC in which each is defined]*
- *[selection: Optional Ciphersuites:*
  - *[assignment: list of optional ciphersuites and reference to RFC in which each is defined];*

  no other ciphersuite]]].

Application Note 185

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. Note that TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP/FW cPP v2.0, but is required to ensure compliance with RFC 5246.

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

Application Note 186

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.

FCS_TLSC_EXT.2.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 187

Validity is determined by the identifier verification, certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

FCS_TLSC_EXT.2.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [assignment: *list of supported curves including an option for "none"*].

Application Note 188

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_TLS_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/SigGen and FCS_CKM.1 and FCS_CKM.2. This extension is required for clients supporting Elliptic Curve ciphersuites.

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

Application Note 189

The use of X.509v3 certificates for TLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include the client must be capable of presenting a certificate to a TLS server for TLS mutual authentication.

FCS_TLSS_EXT TLS Server Protocol 1 2

C.2.2.8 FCS_TLSS_EXT TLS Server Protocol

Family Behaviour

The component in this family addresses the ability for a server to use TLS to protect data between a client and the server using the TLS protocol.

Component levelling

FCS_TLSS_EXT.1 TLS Server requires that the server side of TLS be implemented as specified.

FCS_TLSS_EXT.2: TLS Server requires the mutual authentication be included in the TLS implementation.

Management: FCS_TLSS_EXT.1, FCS_TLSS_EXT.2

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

a) There are no management activities foreseen.

Audit: FCS_TLSS_EXT.1, FCS_TLSS_EXT.2

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

a) Failure of TLS session establishment
b) TLS session establishment
c) TLS session termination

FCS_TLSS_EXT.1	**TLS Server Protocol**

Hierarchical to: No other components

Dependencies:
FCS_CKM.1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_TLSS_EXT.1.1 The TSF shall implement [selection: *TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TSF shall support the following ciphersuites: [selection:

- *Mandatory Ciphersuites:*
  - *[assignment: list of mandatory ciphersuites and reference to RFC in which each is defined]*
- *[selection: Optional Ciphersuites:*
  - *[assignment: list of optional ciphersuites and reference to RFC in which each is defined];*

  no other ciphersuite]]].

Application Note 190

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. Note that TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0, but is required in order to ensure compliance with RFC 5246.

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

Application Note 191

All SSL versions and TLS v1.0 are denied. Any TLS versions not selected in FCS_TLSS_EXT.1.1 should be selected here. (If "none" is the selection for this element then the ST author may omit the words "and none".)

FCS_TLSS_EXT.1.3 The TSF shall generate key establishment parameters using RSA with key size [selection: *2048 bits, 3072 bits, 4096 bits, no other size*] and [selection: *[assignment: List of elliptic curves]; [assignment: list of Diffie-Hellman parameter sizes]*].

Application Note 192

The assignments will be filled in based on the assignments performed in FCS_TLSS_EXT.1.1.

FCS_TLSS_EXT.2	**TLS Server Protocol with mutual authentication**

Hierarchical to: FCS_TLSS_EXT.1 TLS Server Protocol

Dependencies: FCS_CKM.1 Cryptographic Key Generation
FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_TLSS_EXT.2.1 The TSF shall implement [selection: *TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TSF shall support the following ciphersuites: [selection:

- *Mandatory Ciphersuites:*
  - *[assignment: list of mandatory ciphersuites and reference to RFC in which each is defined]*
- *[selection: Optional Ciphersuites:*
  - *[assignment: list of optional ciphersuites and reference to RFC in which each is defined];*

  no other ciphersuite]]].

Application Note 193

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. Note that TLS_RSA_WITH_AES_128_CBC_SHA is not mandatory for ND cPP v2.0, but is required in order to ensure compliance with RFC 5246.

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

Application Note 194

All SSL versions and TLS v1.0 are denied. Any TLS versions not selected in FCS_TLSS_EXT.1.1 should be selected here. (If "none" is the selection for this element then the ST author may omit the words "and none".)

FCS_TLSS_EXT.2.3 The TSF shall generate key establishment parameters using RSA with key size [selection: *2048 bits, 3072 bits, 4096 bits, no other size*] and [selection: *[assignment: list of elliptic curves]; [assignment: list of Diffie-Hellman parameter sizes]*].

Application Note 195

The assignments will be filled in based on the assignments performed in FCS_TLSS_EXT.2.1.

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

Application Note 196

The use of X.509v3 certificates for TLS is addressed in FIA_X509_EXT.2.1. This requirement adds that this use must include support for client-side certificates for TLS mutual authentication.

FCS_TLSS_EXT.2.5 The TSF shall not establish a trusted channel if the client certificate is invalid. If the client certificate is deemed invalid, then the TSF shall [selection: *not establish the connection, request authorization to establish the connection, [assignment: other action]*].

Application Note 197

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

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.

Application Note 198

This requirement only applies to those TOEs performing mutually-authenticated TLS (FCS_TLSS_EXT.2.4). The peer identifier may be in the Subject field or the Subject Alternative Name extension of the certificate. The expected identifier may either be configured, may be compared to the Domain Name, IP address, username, or email address used by the peer, or may be passed to a directory server for comparison.

C.3 Firewall (FFW)

C.3.1 Stateful Traffic Filter Firewall (FFW_RUL_EXT)

Family Behaviour

This requirement is used to specify the behavior of a Stateful Traffic Filter Firewall. The network protocols that the TOE can filter, as well as the attributes that can be used by an administrator to construct a ruleset are identified in this component. How the ruleset is processed (i.e., ordering) is specified, as well as any expected default behavior on the part of the TOE.

Component leveling

FFW_RUL_EXT.1 Stateful Traffic Filtering requires the TOE to filter network traffic based on a ruleset configured by an authorized administrator.

Management: FFW_RUL_EXT.1

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

a) enable/disable a ruleset on a network interface
b) configure a ruleset
c) specifying rules that govern the use of resources

Audit: FFW_RUL_EXT.1

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

a) Minimal:
   - Result (i.e., drop, allow) of applying a rule in the ruleset to a network packet
   - Configuration of the ruleset
   - Indication of packets dropped due to too much network traffic

C.3.1.1 FFW_RUL_EXT.1 Stateful Traffic Filtering

FFW_RUL_EXT.1	**Stateful Traffic Filtering**

Hierarchical to: No other components

Dependencies: None

FFW_RUL_EXT.1.1 The TSF shall perform Stateful Traffic Filtering on network packets processed by the TOE.

FFW_RUL_EXT.1.2 The TSF shall allow the definition of Stateful Traffic Filtering rules using the following network protocol fields: [assignment: *list of attributes supported by the ruleset*].

FFW_RUL_EXT.1.3 The TSF shall allow the following operations to be associated with Stateful Traffic Filtering rules: permit or drop with the capability to log the operation.

FFW_RUL_EXT.1.4 The TSF shall allow the Stateful Traffic Filtering rules to be assigned to each distinct network interface.

FFW_RUL_EXT.1.5 The TSF shall:

a) accept a network packet without further processing of Stateful Traffic Filtering rules if it matches an allowed established session for the following protocols: [assignment: *list of supported protocols for which state is maintained*] based on the following network packet attributes: [assignment: *list of attributes associated with each of the protocols*].

b) Remove existing traffic flows from the set of established traffic flows based on the following: [selection: *session inactivity timeout, completion of the expected information flow*].

FFW_RUL_EXT.1.6 The TSF shall enforce the following default Stateful Traffic Filtering rules on all network traffic: [assignment: *list of default rules that are applied to network traffic flow*].

FFW_RUL_EXT.1.7 The TSF shall be capable of dropping and logging according to the following rules: [assignment: *list of specific rules that the TOE is capable of enforcing*]

FFW_RUL_EXT.1.8 The TSF shall process the applicable Stateful Traffic Filtering rules in an administratively defined order.

FFW_RUL_EXT.1.9 The TSF shall deny packet flow if a matching rule is not identified.

FFW_RUL_EXT.1.10 The TSF shall be capable of limiting an administratively configured number of [assignment: *rules governing the use of resources*].

FIA_PMG_EXT Password Management 1

C.3.1.2 FFW_RUL_EXT.2 Stateful Filtering of Dynamic Protocols

FFW_RUL_EXT.2	**Stateful Filtering of Dynamic Protocols**

Hierarchical to: No other components

Dependencies: None

FFW_RUL_EXT.2.1 The TSF shall dynamically define rules or establish sessions allowing network traffic to flow for the following network protocols [assignment: *list of supported protocols*].

C.4 Identification and Authentication (FIA)

C.4.1 Password Management (FIA_PMG_EXT)

Family Behaviour

The TOE defines the attributes of passwords used by administrative users to ensure that strong passwords and passphrases can be chosen and maintained.

Component levelling

FIA_PMG_EXT.1 Password management requires the TSF to support passwords with varying composition requirements, minimum lengths, maximum lifetime, and similarity constraints.

Management: FIA_PMG_EXT.1

No management functions.

Audit: FIA_PMG_EXT.1

No specific audit requirements.

C.4.1.1 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1	**Password Management**

Hierarchical to: No other components.

Dependencies: No other components.

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

FIA_UIA_EXT User Identification and Authentication 1

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [selection: *"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [assignment: other characters]*];

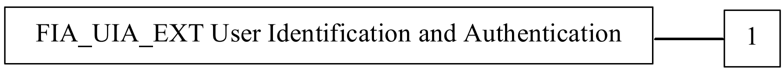
b) Minimum password length shall be settable by the Security Administrator, and support passwords of 15 characters or greater.

C.4.2 User Identification and Authentication (FIA_UIA_EXT)

Family Behaviour

The TSF allows certain specified actions before the non-TOE entity goes through the identification and authentication process.

Component levelling



FIA_UIA_EXT.1 User Identification and Authentication requires Administrators (including remote Administrators) to be identified and authenticated by the TOE, providing assurance for that end of the communication path. It also ensures that every user is identified and authenticated before the TOE performs any mediated functions

Management: FIA_UIA_EXT.1

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

d) Ability to configure the list of TOE services available before an entity is identified and authenticated

Audit: FIA_UIA_EXT.N

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

a) All use of the identification and authentication mechanism
b) Provided user identity, origin of the attempt (e.g. IP address)

C.4.2.1 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1	**User Identification and Authentication**

Hierarchical to:	No other components.
Dependencies:	FTA_TAB.1 Default TOE Access Banners

FIA_UAU_EXT Password-based Authentication Mechanism 2

FIA_UIA_EXT.1.1 The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- [selection: *no other actions, [assignment: list of services, actions performed by the TSF in response to non-TOE requests]*].

FIA_UIA_EXT.1.2 The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

Application Note 199

This requirement applies to users (Administrators and external IT entities) of services available from the TOE directly, and not services available by connecting through the TOE. While it should be the case that few or no services are available to external entities prior to identification and authentication, if there are some available (perhaps ICMP echo) these should be listed in the assignment statement; otherwise "no other actions" should be selected.

Authentication can be password-based through the local console or through a protocol that supports passwords (such as SSH), or be certificate based (such as DTLS, SSH, TLS).

For communications with external IT entities (e.g., an audit server or NTP server, for instance), such connections must be performed in accordance with FTP_ITC.1, whose protocols perform identification and authentication. This means that such communications (e.g., establishing the IPsec connection to the authentication server) would not have to be specified in the assignment, since establishing the connection "counts" as initiating the identification and authentication process.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

C.4.3 User authentication (FIA_UAU_EXT)

Family Behaviour

Provides for a locally based administrative user authentication mechanism

Component levelling

FIA_UAU_EXT Password-based Authentication Mechanism	—	2

FIA_UAU_EXT.2 The password-based authentication mechanism provides administrative users a locally based authentication mechanism.

Management: FIA_UAU_EXT.2

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

a) None

Audit: FIA_UAU_EXT.2

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

a) Minimal: All use of the authentication mechanism

C.4.3.1 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2	**Password-based Authentication Mechanism**

Hierarchical to: No other components.

Dependencies: No other components.

FIA_UAU_EXT.2.1 The TSF shall provide a local password-based authentication mechanism, [selection: *[assignment: other authentication mechanism(s)], none*] to perform administrative user authentication.

Application Note 200

The assignment should be used to identify any additional local authentication mechanisms supported. Local authentication mechanisms are defined as those that occur through the local console; remote administrative sessions (and their associated authentication mechanisms) are specified in FTP_TRP.1/Admin.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

C.4.4 Authentication using X.509 certificates (FIA_X509_EXT)

Family Behaviour

This family defines the behaviour, 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 levelling

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, FIA_X509_EXT.2, FIA_X509_EXT.3

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

a) Remove imported X.509v3 certificates
b) Approve import and removal of X.509v3 certificates
c) Initiate certificate requests

Audit: FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3

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

a) Minimal: No specific audit requirements are specified.

C.4.4.1 FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1	**X.509 Certificate Validation**

Hierarchical to: No other components

Dependencies: FIA_X509_EXT.2 X.509 Certificate Authentication
FIA_X509_EXT.3 X.509 Certificate Requests

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 6960, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5, no revocation method*]
- The TSF shall validate the extendedKeyUsage field according to the following rules: [assignment: *rules that govern contents of the extendedKeyUsage field that need to be verified*].

Application Note 201

FIA_X509_EXT.1.1 lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. If the TOE is distributed and X.509 based authentication is being used to authenticate the protocol selected in FPT_ITT.1, certificate revocation checking is optional. It is optional because there are additional requirements surrounding the enabling and disabling of the FPT_ITT channel defined in FCO_CPC_EXT.1. If revocation is not supported the ST author selects no revocation method. The ST author fills in the assignment with rules that may apply to other requirements in the ST. For instance, if a protocol such as TLS that uses certificates is specified in the ST, then certain values for the extendedKeyUsage field (e.g., "Server Authentication Purpose") could be specified.

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.

Application Note 202

This requirement applies to certificates that are used and processed by the TSF and restricts the certificates that may be added as trusted CA certificates.

C.4.4.2 FIA_X509_EXT.2 X509 Certificate Authentication

FIA_X509_EXT.2	**X.509 Certificate Authentication**

Hierarchical to: No other components

Dependencies: FIA_X509_EXT.1 X.509 Certificate Validation
FIA_X509_EXT.3 X.509 Certificate Requests

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [selection*: DTLS, HTTPS, IPsec, TLS, SSH, [assignment: other protocols], no protocols*], and [selection: *code signing for system software updates, code signing for integrity verification, [assignment: other uses], no additional uses*].

Application Note 203

If the TOE specifies the implementation of communications protocols that perform peer authentication using certificates, the ST author either selects or assigns the protocols that are specified; otherwise, they select "no protocols". Protocols that do not use X.509 based peer authentication include SSH, where ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and/or ecdsa-sha2-nistp521 are selected. The TOE may also use certificates for other purposes; the second selection and assignment are used to specify these cases.

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*].

Application Note 204

Often a connection must be established to check the revocation status of a certificate - either to download a CRL or to perform a lookup using OCSP. The selection is used to describe the behaviour in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate valid according to all other rules in FIA_X509_EXT.1, the behaviour indicated in the selection determines the validity. The TOE must not accept the certificate if it fails any of the other validation rules in FIA_X509_EXT.1. If the Administrator-configured option is selected by the ST Author, the ST Author also selects the corresponding function in FMT_SMF.1.

If the TOE is distributed and FIA_X509_EXT.1/ITT is selected, then certificate revocation checking is optional. This is due to additional authorization actions being performed in the enabling and disabling of the intra-TOE trusted channel as defined in FCO_CPC_EXT.1. In this case, a connection is not required to determine certificate validity and this SFR is trivially satisfied.

C.4.4.3 FIA_X509_EXT.3 X.509 Certificate Requests

FIA_X509_EXT.3	**X.509 Certificate Requests**

Hierarchical to: No other components

Dependencies: FCS_CKM.1 Cryptographic Key Generation
FIA_X509_EXT.1 X.509 Certificate Validation
FIA_X509_EXT.2 X.509 Certificate Authentication

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, [assignment: other information]*].

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

C.5 Protection of the TSF (FPT)

C.5.1 Protection of TSF Data (FPT_SKP_EXT)

Family Behaviour

Components in this family address the requirements for managing and protecting TSF data, such as cryptographic keys. This is a new family modelled after the FPT_PTD Class.

Component levelling

FPT_SKP_EXT Protection of TSF Data — 1

FPT_SKP_EXT.1 Protection of TSF Data (for reading all symmetric keys), requires preventing symmetric keys from being read by any user or subject. It is the only component of this family.

Management: FPT_SKP_EXT.1

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

a) There are no management activities foreseen.

Audit: FPT_SKP_EXT.1

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

a) There are no auditable events foreseen.

C.5.1.1 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all symmetric keys)

FPT_SKP_EXT.1	Protection of TSF Data (for reading of all symmetric keys)

Hierarchical to: No other components.

Dependencies: No other components.

FPT_SKP_EXT.1.1 The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

Application Note 205

The intent of this requirement is for the device to protect keys, key material, and authentication credentials from unauthorized disclosure. This data should only be accessed for the purposes of their assigned security functionality, and there is no need for them to be displayed/accessed at any other time. This requirement does not prevent the device from providing indication that these exist, are in use, or are still valid. It does, however, restrict the reading of the values outright.

FPT_APW_EXT Protection of Administrator Passwords 1

C.5.2 Protection of Administrator Passwords (FPT_APW_EXT)

C.5.2.1 FPT_APW_EXT.1 Protection of Administrator Passwords

Family Behaviour

Components in this family ensure that the TSF will protect plaintext credential data such as passwords from unauthorized disclosure.

Component levelling

FPT_APW_EXT Protection of Administrator Passwords — 1

FPT_APW_EXT.1 Protection of Administrator passwords requires that the TSF prevent plaintext credential data from being read by any user or subject.

Management: FPT_APW_EXT.1

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

a) No management functions.

Audit: FPT_APW_EXT.1

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

a) No audit necessary.

FPT_APW_EXT.1	Protection of Administrator Passwords

Hierarchical to: No other components

Dependencies: No other components.

FPT_APW_EXT.1.1 The TSF shall store passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext passwords.

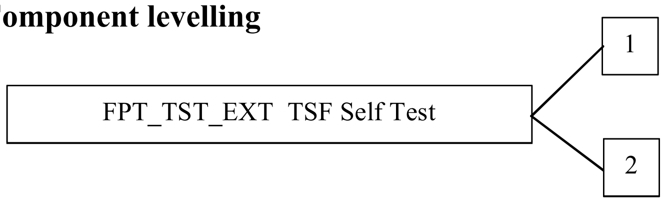
C.5.3 TSF Self-Test (FPT_TST_EXT)

C.5.3.1 FPT_TST_EXT.1 TSF Testing

Family Behaviour

Components in this family address the requirements for self-testing the TSF for selected correct operation.

Component levelling



FPT_TST_EXT.1 TSF Self-Test requires a suite of self-tests to be run during initial start-up in order to demonstrate correct operation of the TSF.

FPT_TST_EXT.2 Self-tests based on certificates applies when using certificates as part of self-test, and requires that the self-test fails if a certificate is invalid.

Management: FPT_TST_EXT.1, FPT_TST_EXT.2

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

a) No management functions.

Audit: FPT_TST_EXT.1, FPT_TST_EXT.2

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

a) Indication that TSF self-test was completed
b) Failure of self-test

FPT_TST_EXT.1	TSF testing

Hierarchical to: No other components.

Dependencies: No other components.

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [selection: *during initial start-up (on power on), periodically during normal operation, at the request of the authorised user, at the conditions [assignment: conditions under which self-tests should occur]*] to demonstrate the correct operation of the TSF: [assignment*: list of self-tests run by the TSF*].

Application Note 206

It is expected that self-tests are carried out during initial start-up (on power on). Other options should only be used if the developer can justify why they are not carried out during initial start-up. It is expected that at least self-tests for verification of the integrity of the firmware and software as well as for the correct operation of cryptographic functions necessary to fulfil the SFRs will be performed. If not all self-tests are performed during start-up multiple iterations of this SFR are used with the appropriate options selected. In future versions of this cPP the suite of self-tests will be required to contain at least mechanisms for measured boot including self-tests of the components which perform the measurement.

FPT_TUD_EXT Trusted Update 1

For distributed TOEs all TOE components have to perform self-tests. This does not necessarily mean that each TOE component has to carry out the same self-tests: the ST describes the applicability of the selection (i.e. when self-tests are run) and the final assignment (i.e. which self-tests are carried out) to each TOE component.

Application Note 207

If certificates are used by the self-test mechanism (e.g. for verification of signatures for integrity verification), certificates are validated in accordance with FIA_X509_EXT.1 and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TST_EXT.2 must be included in the ST.

FPT_TST_EXT.2	**Self-tests based on certificates**

Hierarchical to: No other components.

Dependencies: No other components.

FPT_TST_EXT.2.1 The TSF shall fail self-testing if a certificate is used for self-tests and the corresponding certificate is deemed invalid.

Application Note 208

Certificates may optionally be used for self-tests (FPT_TST_EXT.1.1). This element must be included in the ST if certificates are used for self-tests. If "code signing for integrity verification" is selected in FIA_X509_EXT.2.1, FPT_TST_EXT.2 must be included in the ST.

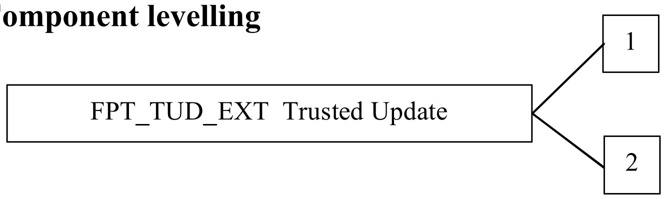
Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with FIA_X509_EXT.1.

C.5.4 Trusted Update (FPT_TUD_EXT)

Family Behaviour

Components in this family address the requirements for updating the TOE firmware and/or software.

Component levelling



FPT_TUD_EXT.1 Trusted Update requires management tools be provided to update the TOE firmware and software, including the ability to verify the updates prior to installation.

FPT_TUD_EXT.2 Trusted update based on certificates applies when using certificates as part of trusted update, and requires that the update does not install if a certificate is invalid.

Management: FPT_TUD_EXT.1

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

a) Ability to update the TOE and to verify the updates
b) Ability to update the TOE and to verify the updates using the digital signature capability (FCS_COP.1/SigGen) and [selection: *no other functions, [assignment: other cryptographic functions (or other functions) used to support the update capability]*]
c) Ability to update the TOE, and to verify the updates using [selection: *digital signature, published hash, no other mechanism*] capability prior to installing those updates

Audit: FPT_TUD_EXT.1, FPT_TUD_EXT.2

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

a) Initiation of the update process.
b) Any failure to verify the integrity of the update

C.5.4.1 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1	Trusted update

Hierarchical to: No other components

Dependencies: FCS_COP.1/SigGen Cryptographic operation (for Cryptographic Signature and Verification), or FCS_COP.1/Hash Cryptographic operation (for cryptographic hashing)

FPT_TUD_EXT.1.1 The TSF shall provide [assignment: *Administrators*] the ability to query the currently executing version of the TOE firmware/software as well as the most recently installed version of the TOE firmware/software.

Application Note 209

The version currently running (being executed) may not be the version most recently installed. For instance, maybe the update was installed but the system requires a reboot before this update will run. Therefore, it needs to be clear that the query should indicate both the most recently executed version as well as the most recently installed update.

FPT_TUD_EXT.1.2 The TSF shall provide [assignment: *Administrators*] the ability to manually initiate updates to TOE firmware/software and [selection: *support automatic checking for updates, support automatic updates, no other update mechanism*].

Application Note 210

The selection in FPT_TUD_EXT.1.2 distinguishes the support of automatic checking for updates and support of automatic updates. The first option refers to a TOE that checks whether a new update is available, communicates this to the Administrator (e.g. through a message during an Administrator session, through log files) but requires some action by the Administrator to actually perform the update. The second option refers to a TOE that checks for updates and automatically installs them upon availability.

The TSS explains what actions are involved in the TOE support when using the "support automatic checking for updates" or "support automatic updates" selections.

When published hash values (see FPT_TUD_EXT.1.3) are used to protect the trusted update mechanism, the TOE must not automatically download the update file(s) together with the hash value (either integrated in the update file(s) or separately) and automatically install the update without any active authorization by the Security Administrator, even when the calculated hash value matches the published hash value. When using published hash values to protect the trusted update mechanism, the option "support of automatic updates" must not be used (automated checking for updates is permitted, though). The TOE may automatically download the update file(s) themselves but not to the hash value. For the published hash approach, it is intended that a Security Administrator is always required to give active authorisation for installation of an update (as described in more detail under FPT_TUD_EXT.1.3) below. Due to this, the type of update mechanism is regarded as "manually initiated update", even if the update file(s) may be downloaded automatically. A fully automated approach (without Security Administrator intervention) can only be used when "digital signature mechanism" is selected in FPT_TUD_EXT.1.3 below.

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [selection: *digital signature mechanism, published hash*] prior to installing those updates.

Application Note 211

The digital signature mechanism referenced in the selection of FPT_TUD_EXT.1.3 is one of the algorithms specified in FCS_COP.1/SigGen. The published hash referenced in FPT_TUD_EXT.1.3 is generated by one of the functions specified in FCS_COP.1/Hash. The ST author should choose the mechanism implemented by the TOE; it is acceptable to implement both mechanisms.

When published hash values are used to secure the trusted update mechanism, an active authorization of the update process by the Security Administrator is always required. The secure transmission of an authentic hash value from the developer to the Security Administrator is one of the key factors to protect the trusted update mechanism when using published hashes and the guidance documentation needs to describe how this transfer has to be performed. For the verification of the trusted hash value by the Security Administrator different use cases are possible. The Security Administrator could obtain the published hash value as well as the update file(s) and perform the verification outside the TOE while the hashing of the update file(s) could be done by the TOE or by other means. Authentication as Security Administrator and initiation of the trusted update would in this case be regarded as "active authorization" of the trusted update. Alternatively, the Administrator could provide the TOE with the published hash value together with the update file(s) and the hashing and hash *comparison is performed by the TOE. In case of successful hash verification, the TOE can perform the update without any additional step by the Security Administrator. Authentication as Security Administrator and sending the hash value to the TOE is regarded as "active authorization" of the trusted update (in case of successful hash verification), because the Administrator is expected to load the hash value only to the TOE when intending to perform the update. As long as the transfer of the hash value to the TOE is performed by the Security Administrator, loading of the update file(s) can be performed by the Security Administrator or can be automatically downloaded by the TOE from a repository.*

If the digital signature mechanism is selected, the verification of the signature shall be performed by the TOE itself. For the published hash option, the verification can be done by the TOE itself as well as by the Security Administrator. In the latter case use of TOE functionality for the verification is not mandated, so verification could be done using non-TOE functionality of the device containing the TOE or without using the device containing the TOE.

For distributed TOEs all TOE components shall support Trusted Update. The verification of the signature or hash on the update shall either be done by each TOE component itself (signature verification) or for each component (hash verification).

Updating a distributed TOE might lead to the situation where different TOE components are running different software versions. Depending on the differences between the different software versions the impact of a mixture of different software versions might be no problem at all or critical to the proper functioning of the TOE. The TSS shall detail the mechanisms that support the continuous proper functioning of the TOE during trusted update of distributed TOEs.

Application Note 212

Future versions of this cPP will mandate the use of a digital signature mechanism for trusted updates.

Application Note 213

If certificates are used by the update verification mechanism, certificates are validated in accordance with FIA_X509_EXT.1 and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TUD_EXT.2 must be included in the ST.

Application Note 214

"Update" in the context of this SFR refers to the process of replacing a non-volatile, system resident software component with another. The former is referred to as the NV image, and the latter is the update image. While the update image is typically newer than the NV image, this is not a requirement. There are legitimate cases where the system owner may want to rollback a component to an older version (e.g. when the component manufacturer releases a faulty update, or when the system relies on an undocumented feature no longer present in the update). Likewise, the owner may want to update with the same version as the NV image to recover from faulty storage.

All discrete firmware and software components (e.g. applications, drivers, and kernel) of the TSF, need to be protected, i.e. they should either be digitally signed by the corresponding manufacturer and subsequently verified by the mechanism performing the update or a hash should be published for them which needs to be verified before the update. Since it is recognized that components may be signed by different manufacturers (in case signatures are used to protect updates), it is essential that the update process verify that both the update and NV *images were produced by the same manufacturer (e.g. by comparing public keys) or signed by legitimate signing keys (e.g. successful verification of certificates when using X.509 certificates).*

C.5.4.2 FPT_TUD_EXT.2 Trusted Update based on certificates

FPT_TUD_EXT.2	**Trusted update based on certificates**

Hierarchical to: No other components

Dependencies: FPT_TUD_EXT.1

FPT_TUD_EXT.2.1 The TSF shall not install an update if the code signing certificate is deemed invalid.

FPT_TUD_EXT.2.2 When the certificate is deemed invalid because the certificate has expired, 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 215

Certificates may optionally be used for code signing of system software updates (FPT_TUD_EXT.1.3). This element must be included in the ST if certificates are used for validating updates. If "code signing for system software updates" is selected in FIA_X509_EXT.2.1, FPT_TUD_EXT.2 must be included in the ST.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with FIA_X509_EXT.1. For expired certificates the author of the ST selects whether the certificate shall be accepted, rejected or the choice is left to the Administrator to accept or reject the certificate.

C.5.5 Time stamps (FPT_STM_EXT)

Family Behaviour

Components in this family extend FPT_STM requirements by describing the source of time used in timestamps.

Component levelling

FPT_STM_EXT.1 Reliable Time Stamps is hierarchic to FPT_STM.1: it requires that the TSF provide reliable time stamps for TSF and identifies the source of the time used in those timestamps.

Management: FPT_STM_EXT.1

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

a) Management of the time
b) Administrator setting of the time.

Audit: FTA_SSL_EXT.1

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

a) Discontinuous changes to the time.

C.5.5.1 FPT_STM_EXT.1 Reliable Time Stamps

FPT_STM_EXT.1	**Reliable Time Stamps**

Hierarchical to: No other components

Dependencies: No other components.

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [selection: *allow the Security Administrator to set the time, synchronise time with an NTP server*].

Application Note 216

Reliable time stamps are expected to be used with other TSF, e.g. for the generation of audit data to allow the Security Administrator to investigate incidents by checking the order of events and to determine the actual local time when events occurred. The decision about the required level of accuracy of that information is up to the Administrator. The TOE depends on external time and date information, either provided manually by the Security Administrator or through the use of one or more external time sources like NTP servers. The corresponding option(s) shall be chosen from the selection in FPT_STM_EXT.1.2. The use of a local real-time clock and the automatic synchronisation with an external time source (e.g. NTP server) is recommended but not mandated. Note that for the communication with an external time source like an NTP server, the use of FTP_ITC.1 is optional but not mandated. The ST author describes in the TSS how the external time and date information is received by the TOE and how this information is maintained.

The term "reliable time stamps" refers to the strict use of the time and date information, that is provided externally, and the logging of all discontinuous changes to the time settings including information about the old and new time. With this information the real time for all audit data can be determined. Note, that all discontinuous time changes, Administrator actuated or changed via an automated process, must be audited. No audit is needed when time is changed via use of kernel or system facilities – such as daytime (3) – that exhibit no discontinuities in time.

For distributed TOEs it is expected that the Security Administrator ensures synchronization between the time settings of different TOE components. All TOE components shall either be in sync (e.g. through synchronisation between TOE components or through synchronisation of different TOE components with external NTP servers) or the offset should be known to the

Administrator for every pair of TOE components. This includes TOE components synchronized to different time zones.

C.6 TOE Access (FTA)

C.6.1 TSF-initiated Session Locking (FTA_SSL_EXT)

Family Behaviour

Components in this family address the requirements for TSF-initiated and user-initiated locking, unlocking, and termination of interactive sessions.

The extended FTA_SSL_EXT family is based on the FTA_SSL family.

Component levelling

FTA_SSL_EXT.1 TSF-initiated session locking, requires system initiated locking of an interactive session after a specified period of inactivity. It is the only component of this family.

Management: FTA_SSL_EXT.1

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

c) Specification of the time of user inactivity after which lock-out occurs for an individual user.

Audit: FTA_SSL_EXT.1

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

b) Any attempts at unlocking an interactive session.

C.6.1.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1	TSF-initiated Session Locking

Hierarchical to: No other components

Dependencies: FIA_UAU.1 Timing of authentication

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [selection:

- *lock the session - disable any activity of the Administrator's data access/display devices other than unlocking the session, and requiring that the Administrator re-authenticate to the TSF prior to unlocking the session;*

- *terminate the session*]

after a Security Administrator-specified time period of inactivity.

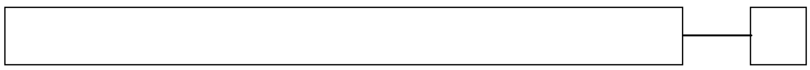
C.7 Communication (FCO)

C.7.1 Communication Partner Control (FCO_CPC_EXT)

Family Behaviour

This family is used to define high-level constraints on the ways that partner IT entities communicate. For example, there may be constraints on when communication channels can be used, how they are established, and links to SFRs expressing lower-level security properties of the channels.

Component levelling



FCO_CPC_EXT.1 Component Registration Channel Definition, requires the TSF to support a registration channel for joining together components of a distributed TOE, and to ensure that the availability of this channel is under the control of an Administrator. It also requires statement of the type of channel used (allowing specification of further lower-level security requirements by reference to other SFRs).

Management: FCO_CPC_EXT.1

No separate management functions are required. Note that elements of the SFR already specify certain constraints on communication in order to ensure that the process of forming a distributed TOE is a controlled activity.

Audit: FCO_CPC_EXT.1

The following actions should be auditable if FCO_CPC_EXT.1 is included in the PP/ST:

a) Enabling communications between a pair of components as in FCO_CPC_EXT.1.1 (including identities of the endpoints).
b) Disabling communications between a pair of components as in FCO_CPC_EXT.1.3 (including identity of the endpoint that is disabled).

If the required types of channel in FCO_CPC_EXT.1.2 are specified by using other SFRs then the use of the registration channel may be sufficiently covered by the audit requirements on those SFRs: otherwise a separate audit requirement to audit the use of the channel should be identified for FCO_CPC_EXT.1.

C.7.1.1 FCO_CPC_EXT.1 Component Registration Channel Definition

FCO_CPC_EXT.1	**Component Registration Channel Definition**

Hierarchical to: No other components.

Dependencies: No other components.

FCO_CPC_EXT.1.1 The TSF shall require a Security Administrator to enable communications between any pair of TOE components before such communication can take place.

FCO_CPC_EXT.1.2 The TSF shall implement a registration process in which components establish and use a communications channel that uses [assignment: *list of different types of channel given in the form of a selection*] for at least [assignment: *type of data for which the channel must be used*].

FCO_CPC_EXT.1.3 The TSF shall enable a Security Administrator to disable communications between any pair of TOE components.

Application Note 217

This SFR is generally applied to a distributed TOE in order to control the process of creating the distributed TOE from its components by means of a registration process in which a component joins the distributed TOE by registering with an existing component of the distributed TOE. When creating the TSF from the initial pair of components, either of these components may be identified as the TSF for the purposes of satisfying the meaning of "TSF" in this SFR.

The intention of this requirement is to ensure that there is a registration process that includes a positive enablement step by an Administrator before components joining a distributed TOE can communicate with the other components of the TOE and before the new component can act as part of the TSF. The registration process may itself involve communication with the joining component: many network devices use a bespoke process for this, and the security requirements for the "registration communication" are then defined in FCO_CPC_EXT.1.2. Use of this "registration communication" channel is not deemed inconsistent with the requirement of FCO_CPC_EXT.1.1 (i.e. the registration channel can be used before the enablement step, but only in order to complete the registration process).

D. Entropy Documentation and Assessment

This appendix describes the required supplementary information for each entropy source used by the TOE.

The documentation of the entropy source(s) should be detailed enough that, after reading, the evaluator will thoroughly understand the entropy source and why it can be relied upon to provide sufficient entropy. This documentation should include multiple detailed sections: design description, entropy justification, operating conditions, and health testing. This documentation is not required to be part of the TSS.

D.1 Design Description

Documentation shall include the design of each entropy source as a whole, including the interaction of all entropy source components. Any information that can be shared regarding the design should also be included for any third-party entropy sources that are included in the product.

The documentation will describe the operation of the entropy source to include how entropy is produced, and how unprocessed (raw) data can be obtained from within the entropy source for testing purposes. The documentation should walk through the entropy source design indicating where the entropy comes from, where the entropy output is passed next, any post-processing of the raw outputs (hash, XOR, etc.), if/where it is stored, and finally, how it is output from the entropy source. Any conditions placed on the process (e.g., blocking) should also be described in the entropy source design. Diagrams and examples are encouraged.

This design must also include a description of the content of the security boundary of the entropy source and a description of how the security boundary ensures that an adversary outside the boundary cannot affect the entropy rate.

If implemented, the design description shall include a description of how third-party applications can add entropy to the RBG. A description of any RBG state saving between power-off and power-on shall be included.

D.2 Entropy Justification

There should be a technical argument for where the unpredictability in the source comes from and why there is confidence in the entropy source delivering sufficient entropy for the uses made of the RBG output (by this particular TOE). This argument will include a description of the expected min-entropy rate (i.e. the minimum entropy (in bits) per bit or byte of source data) and explain that sufficient entropy is going into the TOE randomizer seeding process. This discussion will be part of a justification for why the entropy source can be relied upon to produce bits with entropy.

The amount of information necessary to justify the expected min-entropy rate depends on the type of entropy source included in the product.

For developer-provided entropy sources, in order to justify the min-entropy rate, it is expected that a large number of raw source bits will be collected, statistical tests will be performed, and the min-entropy rate determined from the statistical tests. While no particular statistical tests are required at this time, it is expected that some testing is necessary in order to determine the amount of min-entropy in each output.

For third-party provided entropy sources, in which the TOE vendor has limited access to the design and raw entropy data of the source, the documentation will indicate an estimate of the amount of min-entropy obtained from this third-party source. It is acceptable for the vendor to "assume" an amount of min-entropy, however, this assumption must be clearly stated in the documentation provided. In particular, the min-entropy estimate must be specified and the assumption included in the ST.

Regardless of the type of entropy source, the justification will also include how the DRBG is initialized with the entropy stated in the ST, for example by verifying that the min-entropy rate is multiplied by the amount of source data used to seed the DRBG or that the rate of entropy expected based on the amount of source data is explicitly stated and compared to the statistical rate. If the amount of source data used to seed the DRBG is not clear or the calculated rate is not explicitly related to the seed, the documentation will not be considered complete.

The entropy justification shall not include any data added from any third-party application or from any state saving between restarts.

D.3 Operating Conditions

The entropy rate may be affected by conditions outside the control of the entropy source itself. For example, voltage, frequency, temperature, and elapsed time after power-on are just a few of the factors that may affect the operation of the entropy source. As such, documentation will also include the range of operating conditions under which the entropy source is expected to generate random data. Similarly, documentation shall describe the conditions under which the entropy source is no longer guaranteed to provide sufficient entropy. Methods used to detect failure or degradation of the source shall be included.

D.4 Health Testing

More specifically, all entropy source health tests and their rationale will be documented. This will include a description of the health tests, the rate and conditions under which each health test is performed (e.g., at start up, continuously, or on-demand), the expected results for each health test, TOE behaviour upon entropy source failure, and rationale indicating why each test is believed to be appropriate for detecting one or more failures in the entropy source.

E. Rationales

E.1 SFR Dependencies Analysis

The dependencies between SFRs implemented by the TOE are addressed as follows.

SFR	Dependencies	Rationale Statement
FAU_GEN.1	FPT_STM.1	FPT_STM_EXT.1 included (which is hierarchic to FPT_STM.1)
FAU_GEN.2	FAU_GEN.1 FIA_UID.1	FAU_GEN.1 included Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator identification timing
FAU_STG_EXT.1	FAU_GEN.1 FTP_ITC.1	FAU_GEN.1 included FTP_ITC.1 included
FCS_CKM.1	FCS_CKM.2 or FCS_COP.1 FCS_CKM.4	FCS_CKM.2 included FCS_CKM.4 included
FCS_CKM.2	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1 FCS_CKM.4	FCS_CKM.1 included (also FTP_ITC.1 as a secure channel that could be used for import) FCS_CKM.4 included
FCS_CKM.4	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1	FCS_CKM.1 included (also FTP_ITC.1 as a secure channel that

		could be used for import)
FCS_COP.1/DataEncryption	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1 FCS_CKM.4	FCS_CKM.1 included (also FTP_ITC.1 as a secure channel that could be used for import) FCS_CKM.4 included
FCS_COP.1/SigGen	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1 FCS_CKM.4	FCS_CKM.1 included (also FTP_ITC.1 as a secure channel that could be used for import) FCS_CKM.4 included
FCS_COP.1/Hash	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1 FCS_CKM.4	This SFR specifies keyless hashing operations, so initialisation and destruction of keys are not relevant
FCS_COP.1/KeyedHash	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1 FCS_CKM.4	FCS_CKM.1 included (also FTP_ITC.1 as a secure channel that could be used for import) FCS_CKM.4 included
FCS_RBG_EXT.1	None	
FDP.RIP.2	None	
FIA_AFL.1	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant

		Administrator authentication
FIA_PMG_EXT.1	None	
FIA_UIA_EXT.1	FTA_TAB.1	FTA_TAB.1 included
FIA_UAU_EXT.2	None	
FIA_UAU.7	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator authentication
FMT_MOF.1/ManualUpdate	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_MTD.1/CoreData	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_SMF.1	None	
FMT_SMR.2	FIA_UID.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator identification
FPT_SKP_EXT.1	None	
FPT_APW_EXT.1	None	
FPT_TST_EXT.1	None	
FPT_TUD_EXT.1	FCS_COP.1/SigGen or FCS_COP.1/Hash	FCS_COP.1/SigGen and FCS_COP.1/Hash included
FPT_STM_EXT.1	None	

FTA_SSL_EXT.1	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator authentication
FTA_SSL.3	None	
FTA_SSL.4	None	
FTA_TAB.1	None	
FTP_ITC.1	None	
FTP_TRP.1/Admin	None	
FFW_RUL_EXT.1	None	

Table 6: SFR Dependencies Rationale for Mandatory SFRs

SFR	Dependencies	Rationale Statement
FAU_STG.1	FAU_STG.3	FAU_STG.3/LocSpace included as optional SFRs
FAU_STG_EXT.2/LocSpace	FAU_GEN.1 FAU_STG_EXT.1	FAU_GEN.1 & FAU_STG_EXT.1 included
FAU_STG.3/LocSpace	FAU_STG.1	FAU_STG.1 included as optional SFR
FIA_X509_EXT.1/ITT	None	
FMT_MOF.1/Service	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_MOF.1/Functions	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_MTD.1/CryptoKeys	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FPT_ITT.1	None	
FTP_TRP.1/Join	None	
FCO_CPC_EXT.1	None	
FFW_RUL_EXT.2	None	

Table 7: SFR Dependencies Rationale for Optional SFRs

SFR	Dependencies	Rationale Statement
FIA_X509_EXT.1/Rev	None	
FIA_X509_EXT.2	None	
FIA_X509_EXT.3	FCS_CKM.1	
FCS_DTLSC_EXT.1	FCS_CKM.1 FCS_CKM.2	FCS_CKM.1 included FCS_CKM.2 included

	FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_DTLSC_EXT.2	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_DTLSS_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_DTLSS_EXT.2	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included

	FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_RBG_EXT.1 included
FCS_HTTPS_EXT.1	FCS_TLSC_EXT.1 or FCS_TLSS_EXT.1	FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 included as selection-based SFRs
FCS_IPSEC_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_SSHC_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_SSHS_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included

	FCS_RBG_EXT.1	
FCS_TLSC_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_TLSC_EXT.2	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_TLSS_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FCS_TLSS_EXT.2	FCS_CKM.1 FCS_CKM.2	FCS_CKM.1 included FCS_CKM.2 included

	FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1	FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included
FPT_TST_EXT.2	None	
FPT_TUD_EXT.2	FPT_TUD_EXT.1	FPT_TUD_EXT.1 included
FMT_MOF.1/AutoUpdate	FMT_SMR.1 FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included

Table 8: SFR Dependencies Rationale for Selection-Based SFRs

Glossary

Term	Meaning
Administrator	See Security Administrator.
Assurance	Grounds for confidence that a TOE meets the SFRs [CC1].
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.
Security Administrator	The terms "Administrator" and "Security Administrator" are used interchangeably in this document at present.
Target of Evaluation	A set of software, firmware and/or hardware possibly accompanied by guidance. [CC1]
TOE Security Functionality (TSF)	A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the SFRs. [CC1]
TSF Data	Data for the operation of the TSF upon which the enforcement of the requirements relies.
User	See Security Administrator

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

Acronyms

Acronym	Meaning
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
CA	Certificate Authority
CBC	Cipher Block Chaining
CRL	Certificate Revocation List
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
DTLS	Datagram Transport Layer Security
ECDH	Elliptic Curve Diffie Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
EEPROM	Electrically Erasable Programmable Read-Only Memory
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
HMAC	Keyed-Hash Message Authentication Code
HTTPS	HyperText Transfer Protocol Secure
IP	Internet Protocol
IPsec	Internet Protocol Security
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
PP	Protection Profile
RBG	Random Bit Generator
RSA	Rivest Shamir Adleman Algorithm
SD	Supporting Document
SHA	Secure Hash Algorithm
SSH	Secure Shell
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSS	TOE Summary Specification
VPN	Virtual Private Network