Common Criteria for Information Technology Security Evaluation

Part 2: Security functional components

July 2005

Version 3.0
Revision 2

CCMB-2005-07-002

Foreword

This version of the Common Criteria for Information Technology Security Evaluation (CC v3.0) is the first major revision since being published as CC v2.1 in 1999 and CC v2.2 in 2004.

CC v3.0 is released for public comment and aims to eliminate redundant evaluation activities; reduce/eliminate those activities that contributed little to the final assurance of a product; clarify CC terminology to reduce misunderstandings; restructure and refocus the evaluation activities to those areas where security assurance would truly be gained; and add new CC requirements if needed.

This revision 2 of the CC v3.0 includes all editorial updates as of the release date.

CC version 3.0 consists of the following parts:

– Part 1: Introduction and general model

– Part 2: Security functional components

– Part 3: Security assurance components

Trademarks:

– Microsoft is a registered trademark of Microsoft Corporation

– POSIX is a registered trademark of the IEEE

– UNIX is a registered trademark of The Open Group in the United States and other countries

– Windows is a registered trademark of Microsoft Corporation in the United States and other countries

Legal Notice:

The governmental organisations listed below contributed to the development of this version of the Common Criteria for Information Technology Security Evaluation. As the joint holders of the copyright in the Common Criteria for Information Technology Security Evaluation, version 3.0 *Parts 1 through 3 (called "CC* 3.0*"), they hereby grant non-exclusive license to ISO/IEC to use CC* 3.0 *in the continued development/maintenance of the ISO/IEC 15408 international standard. However, these governmental organisations retain the right to use, copy, distribute, translate or modify CC* 3.0 *as they see fit.*

Australia/New Zealand:	*The Defence Signals Directorate and the Government Communications Security Bureau respectively;*
Canada:	*Communications Security Establishment;*
France:	*Direction Centrale de la Sécurité des Systèmes d'Information;*
Germany:	*Bundesamt für Sicherheit in der Informationstechnik;*
Japan:	*Information Technology Promotion Agency*
Netherlands:	*Netherlands National Communications Security Agency;*
Spain:	*Ministerio de Administraciones Públicas and Centro Criptológico Nacional;*
United Kingdom:	*Communications-Electronics Security Group;*
United States:	*The National Security Agency and the National Institute of Standards and Technology.*

Table of Contents

Table of contents

List of figures

List of tables

1 Introduction

1 Security functional components, as defined in this CC Part 2, are the basis for the SFRs (security functional requirements) expressed in a Protection Profile (PP) or a Security Target (ST). These SFRs describe the desired security behaviour of a Target of Evaluation (TOE) and are intended to meet the security objectives for the TOE as stated in a PP or an ST.

2 The audience for this CC Part 2 includes consumers, developers, and evaluators of IT products. CC Part 1 Chapter 6.3 provides additional information on the target audience of the CC, and on the use of the CC by the groups that comprise the target audience. These groups may use this part of the CC as follows:

- Consumers, who use this CC Part 2 when selecting components to express SFRs to satisfy the security objectives for the TOE expressed in a PP or ST. CC Part 1 Annex A provides more detailed information on the relationship between security objectives and SFRs.

- Developers, who respond to actual or perceived consumer security requirements, may find a standardised method to formulate those requirements in this part of the CC.

- Evaluators, who verify that the SFRs expressed in the PP or ST satisfy the security objectives for the TOE in that PP or ST and that all dependencies are accounted for.

2 Scope

3 This part of the CC defines the required structure and content of security functional components for the purpose of security evaluation. It includes a catalogue of functional components that will meet the common security functionality requirements of many IT products.

3 Normative references

4 The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

CC Common Criteria for Information Technology Security Evaluation, Version 3.0, revision 2, June 2005. Part 1: Introduction and general model.

4 Terms and definitions, symbols and abbreviated terms

5 For the purposes of this document, the terms, definitions, symbols and abbreviated terms given in CC Part 1 apply.

5 Overview

6 The security functional components described in this CC Part 2 are not meant to be a definitive answer to all the problems of IT security. Rather, the CC offers a set of well understood security functional components that may be used to create SFRs reflecting the needs of the market. These security functional components are presented as the current state of the art in requirements specification and evaluation.

7 This part of the CC does not presume to include all possible security functional components but rather contains those that are known by the CC Part 2 author at the time of release.

8 Since the understanding and needs of consumers may change, it is envisioned that some PP/ST authors may have security needs not covered by CC Part 2. In those cases the PP/ST author may choose to consider defining new functional components (referred to as extended components), as explained in Annexes A and B of CC Part 1.

5.1 Organisation of CC Part 2

9 Chapter 6 describes the paradigm used in the security functional components of CC Part 2.

10 Chapter 7 introduces the catalogue of CC Part 2 functional components and describes the structure and presentation of these components.

11 Chapter 8 summarises the six functional classes: major groups of functional components that share a general theme, and Chapters 9 through 14 describe each functional class in detail.

12 Annex A provides a complete cross reference table of the functional component dependencies.

6 Functional paradigm

13 This section describes the paradigm used in this CC Part 2. This section is focused at understanding CC Part 2 and is not intended to replace or supersede any of the terms found in the CC glossary in CC Part 1, Section 4.

6.1 The TSP

14 The set of all SFRs in a given PP/ST is collectively referred to as the TSP (TOE Security Policy).

15 During a TOE evaluation it will be determined whether there exists a predefined level of assurance that the TOE meets the TSP in the Operational Environment. If this level of assurance is met, the evaluation succeeds, otherwise the evaluation fails.

16 It is therefore imperative that the TSP is well-defined. To this end, the CC mandates basing the TSP on the components defined in this CC Part 2 where possible. For exceptions to this case, see CC Part 1, Annex C.5.

17 The TSP (the set of all SFRs in a PP/ST) should be seen as an abstract, implementation independent, specification of the required security behaviour of the TOE:

a) *Abstract*, because it describes the expected behaviour using abstract entities (objects, subjects etc.) rather than entities such as disks, data buses etc.

b) *Implementation-independent*: there may exist many TOEs that meet the TSP but these TOEs may be implemented in a completely different way.

6.2 Subjects, objects and operations

18 The security functional components that are defined in this CC Part 2 use a common set of concepts. This section will explain those concepts.

19 A *subject* is an active entity in the TOE: subjects perform operations and actions in the TOE. A typical implementation of a subject is a kernel process. Subjects are distinct from active entities outside the TOE (users).

20 Note that the CC does not limit subjects to the traditional OS-based shells: any process can be a subject, such as:

– a assembly code interpreter (for a smartcard integrated circuit)

– a network input handler (for a firewall)

– a printer (for a large IT-system)

21 An *object* is a passive entity in the TOE: they are the entities that subjects perform operations on. Typical implementations of objects are a file, a queue and a database.

22 It is possible for an entity to be both a subject and an object. An example of this would be a process that may read from some file, but may itself be terminated by another process.

23 An *operation* is a specific type of action of a subject to an object. An example of an operation is “modify”. Actions that brings an object into being (e.g. the operation “create” is also considered to be an operation, though strictly speaking the object does not yet exist.

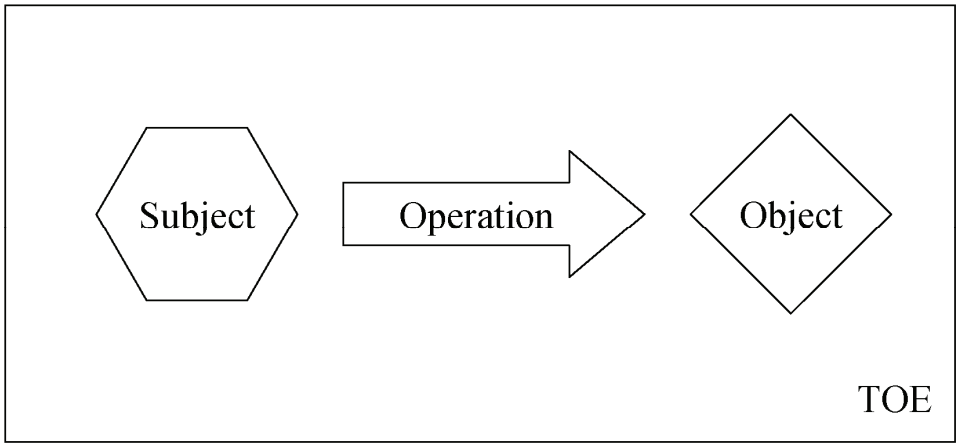


Figure 1 - Subjects, objects and operations

6.3 Choosing subjects, objects and operations in a PP/ST

24 The CC does not mandate a particular choice for subjects, objects and operations. This choice is left to the PP/ST author. The CC also does not mandate any specific choice of granularity for these subjects, objects and operations.

25 For a given type of TOE, such as an OS/workstation combination, a PP/ST author may therefore choose between a variety of objects, subjects, and operations to model. For storage objects, the author could choose to model:

- Storage entities
- Individual files
- Named groups of files
- Types of files
- Hard disk sectors
- Individual bits on a hard disk

or any other choice in between (or more detailed or more abstract) or any combination of these choices (e.g. modelling files and disk sectors).

26 The PP/ST author should choose the level of abstraction based on:

a) The level of detail of the security objectives for the TOE in the PP or ST. If these objectives are very detailed, the SFRs will have to use detailed objects, subjects and operations, otherwise it will be very difficult to map objectives and SFRs to each other.

b) The level of detail in the functional specification (in the case of a TOE evaluation). If this functional specification is abstract, the SFRs will have to use abstract objects, subjects and operations, otherwise it will be very difficult to map the functional specification to the SFRs. In general, this means that only entities that are readily apparent at the functional specification level of abstraction should be modelled, and that many internal structures that are invisible at the functional specification level of abstraction should not be present in the TSP.

27 The PP/ST author may choose to model at any level of granularity as long as the resulting TSP is coherent and meets the applicable Security requirements (ASE_REQ) criteria.

28 For a very simple operating system/workstation, the definition of subjects, objects and operations could look like this:

– Subjects: shell

– Objects: files

– Operations: read, write, create, destroy, print

29 Note that in the implementation of the operating system/workstation, there will be a lot more:

a) “passive entities” such as disk sectors, memory areas, CPU caches, printer queues;

b) “active entities” such as memory managers, process schedulers, printer daemons;

c) “actions” such as initialise_printer_queue, initialise_memory, query_printer_daemon, defragment_disk etc.

but only the entities **defined** to be subjects, objects and operations are subjects, objects and operations.

30 SFRs such as access control do not apply directly to entities that are not defined as objects, subjects and operations. However, the SFRs do apply to these entities indirectly. Specifically, the implementation shall be a valid instantiation of the TSP. This is best illustrated with some examples:

a) If the TSP states that a certain subject is not allowed to execute the modify operation on a certain file, and the implementation has various functions to edit individual disk sectors, that subject should not be able to access those functions, even though those functions are not directly mentioned in the TSP.

b) If the TSP states that a certain subject is not allowed to execute the read operation on a certain file, and another subject prints that file, the first subject should not be able to obtain the contents of that file from the printer queue, even though the printer queue is not mentioned in the TSP.

31 Determining whether the implementation is actually a valid instantiation of the TSP is handled in the ADV: Development, ATE: Tests and AVA: Vulnerability assessment classes.

6.4 Security attributes

32 Subjects and objects may have many properties, not all of which are relevant to the secure operation of the TOE. The properties that are relevant are modelled by security attributes. Security attributes have values, and these values are used in enforcing the TSP. Summarising: security attributes are properties of objects and subjects that are used in defining the TSP and whose values are used in enforcing the TSP.

33 Examples of security attributes are:

a) *Identity attributes:* These represent the identity of an object or subject, such as the name of a file;

b) *Time attributes:* These may be used to specify that certain operations will be authorised during certain times of the day or during certain days of the week;

c) *Location attributes:* These may be used to specify the location of the subject in the TOE, or the location in the TSF where the operation will be carried out, or both;

d) *Role attributes:* These may be used to specify the role the subject is currently performing in the TOE (e.g. regular_user, administrator).

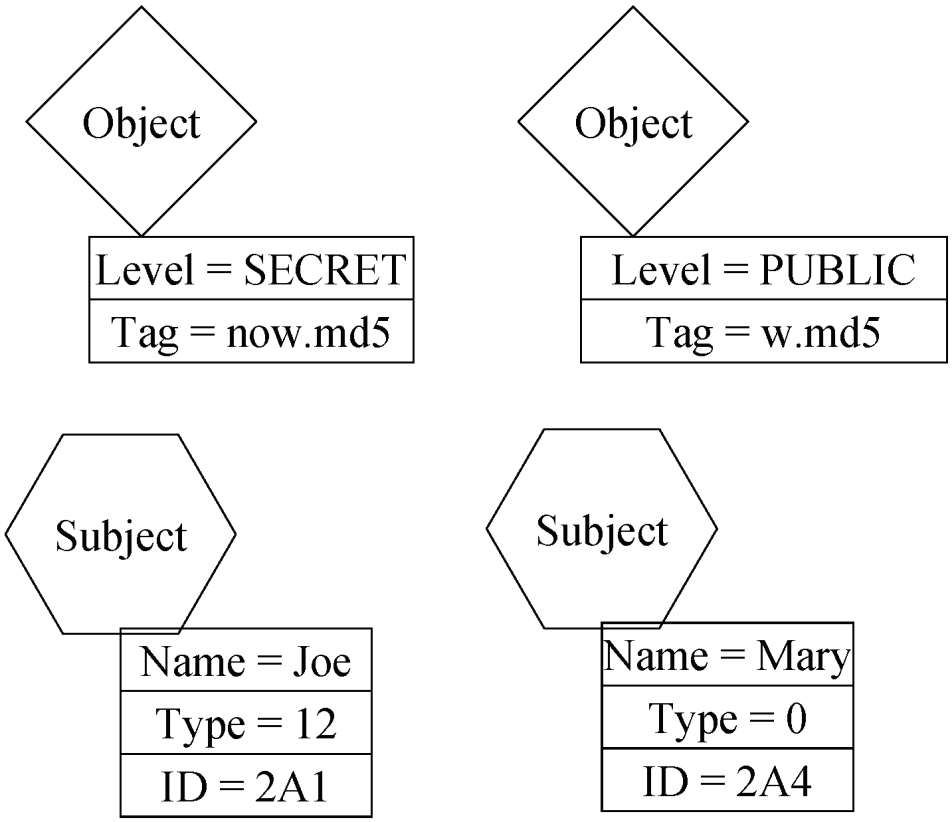


Figure 2 - Some example objects and subjects with security attributes

34 At the level of abstraction of the TSP, a security attribute is merely a tag with a value. This tag may be implemented in various ways on the actual TOE. E.g., an object may have a security attribute “owner” with the value “Joe”. In a TOE this could be implemented as:

a) a file with a tag “Joe”;

b) a file in the directory “Joe”;

c) a file which has the filename <xyz>.Joe etc.

35 The values of security attributes may have interrelations, such as:

a) not ordered: there is no explicit relation between the different values such that one value is considered to be “higher” than the others: the values are incomparable. An example is a security attribute “colour” having values [red, green, blue].

b) ordered: there is a relation between the different values, such that it is always possible to determine which one is the “highest” of the two. Examples are:

– a security attribute “alarm-status” having values [green, yellow, red],

- a security value “number” having values [1, 2, 3, ...],
- a security attribute “classification level” having values [unclassified, confidential, secret, top secret]

c) partially ordered: there is a relation between some of the values, while some other values are incomparable. An example is a security attribute “project classification” having the values

- [Unclassified]
- [Project_Alpha; Confidential]
- [Project_Alpha; Secret]
- [Project_Beta; Confidential]
- [Project_Beta; Secret]

[Project_Alpha; Secret] is greater than [Project_Alpha; Confidential] but is not comparable with [Project_Beta; Secret].

36 If these (or other) relations are used in the TSP (e.g. in Access control (FDP_ACC)), these relations shall be clearly specified in the definition of the security attributes as required by the Security requirements (APE_REQ) and Security requirements (ASE_REQ) families in CC Part 3.

6.5 Users

37 A user is any active entity outside of the TOE. It is important to realise that the CC does not limit users to the traditional concept of human users. In the CC a user may be a human or a machine.

38 Typical machine users are: an application running on an OS TOE, an OS supporting an application TOE, an intrusion detection system feeding an audit analyser TOE across the Internet.

39 In the context of a particular evaluation, an active entity is either a user (outside the TOE) or a subject (inside the TOE), but never both.

40 Users may have *user security properties* associated with them. These user security properties are usually initialised during a user registration process, usually stored in the TOE, and when that user binds (see the next section) to a subject, these user security properties are translated or transferred into values for the security attributes of that subject. This process is described in more detail in the FIA class.

6.6 Bindings

41 In the CC, the only way for users to communicate with the TOE is by communicating with subjects (through interfaces). Users shall not communicate with objects directly.

42 If a user wishes to communicate with an object, he shall do this through a subject, which, in turn, communicates with the object by performing operations on it. In order to communicate with a subject, users shall first associate themselves with that subject, through a process called *binding*.

43 Binding is the CC term used to describe the general process of associating a user with a subject. This process has many parameters, such as:

– whether users are allowed to bind to the subject at all;

– whether users need to identify and/or authenticate themselves as part of the binding process;

– whether the subject being bound to needs to authenticate itself to the user;

– whether the security attributes of the subject change as a result of the binding, to allow the subject to execute certain operations, depending on the user that is bound to it.

For a more complete list, see the FIA: Identification, Authentication and Binding class.

44 Bindings can therefore take many forms:

a) very simple: an anonymous user simply sends data to and receives data from a subject;

b) complex: a user identifies and authenticates itself, and, as a result, the subject gains additional rights allowing it to perform more operations.

45 Once bound to a subject, users may send data to that subject, receive data from that subject or direct the subject to do things. The binding may be long or short in duration. Examples of bindings are:

a) a user sends a single UDP packet to the WAN-Input subject in a firewall. The binding exists only for a brief moment;

b) a user logs in (binds) to a GUI subject in an Operating System, has a session of several hours and logs off.

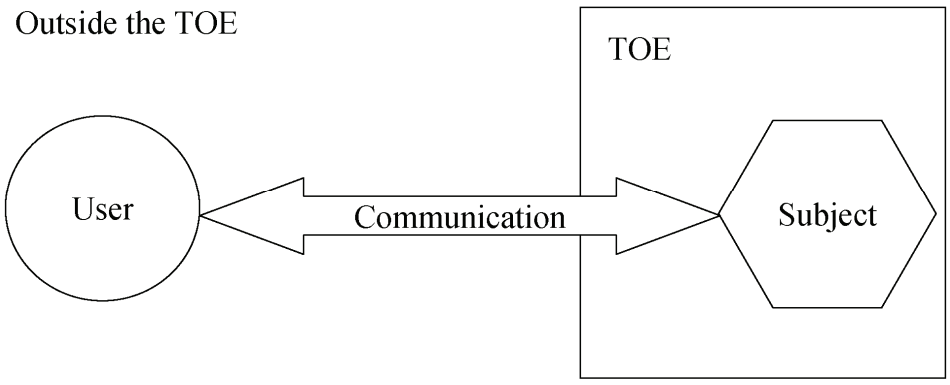


Figure 3 - A user binding to a subject

46 Users shall only communicate with subjects:

a) that they are bound to, or

b) to set up a binding

47 Users shall never directly communicate with objects, as an object is passive. Only an active entity in the TOE (i.e. a subject) may interact with that object. Users shall never directly communicate with subjects they are not bound to (except to bind with them).

48 Note that subjects do not have to be bound to users to perform operations; they just have to be bound to users if these users wish to communicate with the subjects and/or direct the subjects to perform operations. Also note that the same user may be bound to multiple subjects, and that multiple users may be bound to the same subject.

6.7 Notation Conventions

49 It is important to note that the TSP only distinguishes between different objects and subjects based on the values of their security attributes. This could lead to unwieldy notations if applied strictly, such as “An object where the security attribute Name has the value Accounts and the security attribute Type has the value File”.

50 As long as the PP/ST author clearly explains this in the PP/ST, the PP/ST author may use various notational conventions to enhance the readability of the PP/ST, such as:

a) Object<File,Accounts>, O.FILE.ACCOUNTS, The File “Accounts”, the accounts file (for the example given above);

b) A file (for an object where the security attribute Type has the value File)

c) All files (for all objects where the security attribute Type has the value File) etc.

However, this notation should be non-ambiguous and, where necessary, be clearly explained in the PP/ST.

51 Another example would be where an assignment needs to be completed with a "list of objects". This could be completed with a list of all objects that it applies to, but this list may be represented as:

a) a list [file1, file2, file3,..., file100]

b) a set (all files)

c) a qualified list (all files that may be accessed by subject Y) etc.

6.8 The TSF

52 A TSF (TOE Security Functionality) is a part of the TOE (it may be the whole TOE). For hardware it would be a physical part of the hardware, while for software it would be a part of the run-time executable code.

53 In order for a part of the TOE to be a TSF it shall have the following property:

a) if the TSF works as specified, and

b) if all the guidance is followed correctly, and

c) if all the security objectives for the operational environment are met

d) then the TOE will meet the TSP, even if the non-TSF part of the TOE is actively misbehaving and hostile.

54 Therefore, the TSF is strongly tied to the TSP. If for a given TOE the TSP changes, whether a given part of that TOE constitutes a TSF or not may change as well.

55 There may be multiple choices for a TSF for a given TOE and TSP. If any part of a TOE is a TSF, any part of that TOE containing that part is also a TSF.

56 There is therefore a choice in determining the TSF for a given evaluation. In the CC, the developer chooses a part of his TOE (or the entire TOE) as the TSF. If this choice is invalid (the TSF is too small), this will probably cause the evaluation to fail as it will become impossible to meet certain SARs.

6.8.1 Why have a TSF?

57 The reasons that the CC makes a distinction between TSF and TOE are that:

a) The evaluator does not need to examine non-TSF parts of the TOE, because these cannot interfere with the TSF.

b) If the evaluator gains assurance in that the TSF meets the TSP, the evaluator automatically gains equal assurance that the TOE meets the TSP.

58 Consider the following example: the TOE is some configuration of an OS consisting of a kernel and a few hundred applications. If this TOE was evaluated without the TSF concept, the documentation required by all assurance components in the ST (mainly those in ADV) would have to describe the kernel and all applications. However, by using the fact that applications are unable to access the kernel or other applications directly, and that perhaps only a few out of the few hundred applications are needed to implement the TSP, one could construct a TSF consisting of the kernel and those few applications.

59 This would strongly limit the volume of documentation required by ADV: Development, the amount of work needed to satisfy the process requirements required by ALC: Life-cycle support, and the testing effort required by ATE: Tests and AVA: Vulnerability assessment. Additionally, the developer could focus all his security design efforts on that kernel, hopefully resulting in less work and a more robust kernel.

6.8.2 The TSF as an object or subject

60 In the CC, the TSF may be an object: that is, the setting of certain parameters may be modelled as an operation that has the TSF as object.

61 However, the TSF is not a subject: users cannot bind to the TSF itself, as they can only bind to subjects.

6.9 On distributed TOEs

62 This CC Part 2 was written as implementation-independent as possible. This means that there may exist many TOEs that meet the TSP but these TOEs may be implemented in a completely different way. A particular example is a TOE that is implemented in a distributed manner, e.g. a client-server TOE communicating across the Internet.

63 As “being distributed” is not a security requirement but rather an implementation choice, this CC Part 2 has no special components to model this.

64 A TSP does not therefore need to describe how distributed components of a TOE “bind” to themselves, how communication between these parts is protected, etc. as these are implied by the other SFRs.

65 As an example, if the TSF itself is distributed across the Internet, and there is no protection of the communication between TSF parts and no mutual authentication between TSF parts, either by the TSF or by the Operational Environment, the developer may encounter substantial difficulties in implementing other SFRs.

66 Those authors who still want to specify in more detail how a distributed TOE should be protected, may consult section 6.10.5.

6.10 Algorithms, mechanisms and external standards

67 This CC Part 2 was written as implementation-independent as possible. This means that SFRs based on this CC Part 2 may be implemented in many ways. However, in some cases, the PP/ST author may wish to mandate that certain SFRs shall be implemented using specific algorithms, specific mechanisms, specific external standards etc.

68 Rather than creating extended requirements for these implementation-dependent constructs, the PP/ST author should use refinements to tailor the applicable SFRs to the specific problem. A number of examples follows. Many of these examples use refinements to improve readability.

6.10.1 Specifying the use of a particular biometric mechanism for authentication

FIA_UAU.1 **User authentication by TSF**

FIA_UAU.1.1 The TSF shall authenticate a user **by fingerprint recognition** before the user can bind to the **General-User-Shell**.

6.10.2 Specifying key generation for a particular algorithm

FMI_RND.1 **Random number generation**

FMI_RND.1.1 The TSF shall generate **keys** that **consist of 56 random bits, of which no single bit can be predicted with more than 51% probability, and which are not weak or semi-weak DES keys.**

FMI_RND.1.2 The TSF shall store these **keys** in **KeyArea**.

6.10.3 Specifying the use of cryptography at various level of detail

FCO_CED.1 **Confidentiality of exported data**

FCO_CED.1.1 The TSF shall **use cryptography** to protect the confidentiality of **all data** provided by **ftp_handler** to a user bound to **ftp_handler**.

69 or, alternatively

FCO_CED.1 **Confidentiality of exported data**

FCO_CED.1.1 The TSF shall **use AES-256** to protect the confidentiality of **all data** provided by **ftp_handler** to a user bound to **ftp_handler**.

70 or, alternatively

FCO_CED.1 **Confidentiality of exported data**

FCO_CED.1.1 The TSF shall **use AES-256 in accordance with FIPS-197** to protect the confidentiality of **all data** provided by **ftp_handler** to a user bound to **ftp_handler**.

6.10.4 Specifying the use of cryptography as a service

71 This is more complex than the previous examples. The PP/ST author should first define the cryptographic algorithms as operations. The PP/ST author should then specify the subject(s) that can perform these operations.

72 Subsequently, the PP/ST author should define the requirements for the handling of the key, the clear text and the cipher text, by using Access control (FDP_ACC).

73 These requirements can then be extended with FIA: Identification, Authentication and Binding requirements to handle binding between users and the subject, and FCO: Communication requirements to handle confidentiality and integrity between users and the subject.

74 An example is given below. To keep this example understandable, it is purposely incomplete, and, when used in a real PP or ST would require additional SFRs.

FDP_ACC.1 **Access control**

FDP_ACC.1.1 The TSF shall **allow** an operation of a subject on an object **if and only if**

a) **Crypto_Handler performs AES-encryption on (clear text, key, cypher text)**

b) **Crypto_Handler performs AES-decryption on (cypher text, key, clear text)**

FIA_UAU.1 **User authentication by TSF**

FIA_UAU.1.1 The TSF shall authenticate a user before the user can bind to **Crypto_Handler**.

FIA_UAU.2 **User identification**

FIA_UID.2.1 The TSF shall identify a user before the user can bind to **Crypto_Handler**.

6.10.5 Specifying cryptography for internal protection

75 Even though it is strictly not necessary (see section 6.9), some authors may want to specify the specific cryptographic measures to be taken for “internal” protection of a TOE that is likely to be implemented in a distributed manner (e.g. a client-server application). In this case the author could define two subjects “Client” and “Server” and define operations between those subjects such as: “Send_To_Client_Using_3DES” (where the definition of these operations can go in even more detail if so desired).

7 Security functional components

7.1 Security functional classes, families and components structure

76 This chapter defines the content and presentation of the functional components of the CC. To enhance understanding, the functional components are grouped in classes and families.

7.1.1 Functional class structure

77 Figure 4 illustrates the functional class structure in diagrammatic form. Each functional class includes a class name, a class introduction, and one or more functional families.

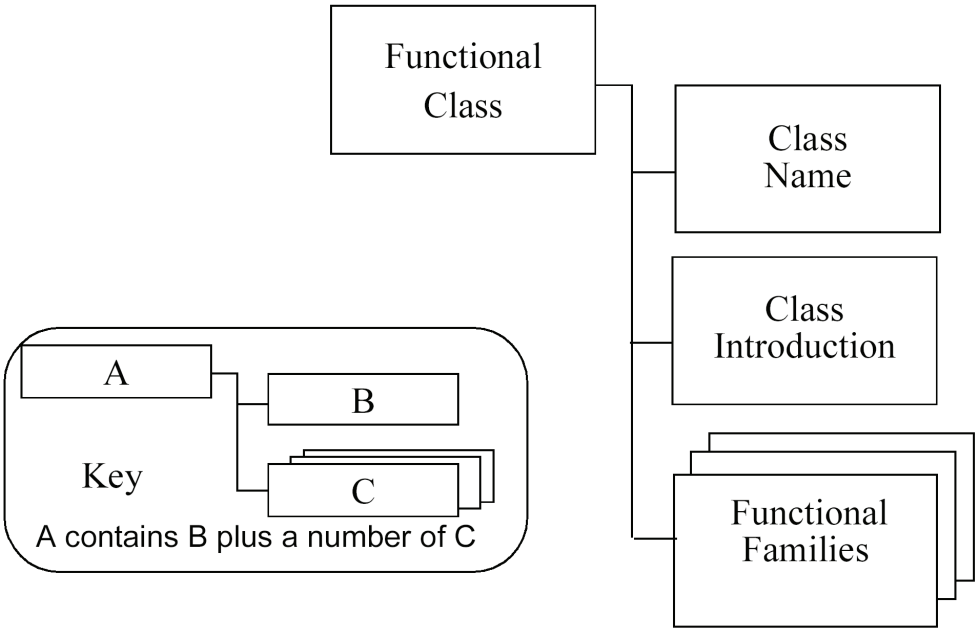


Figure 4 - Functional class structure

7.1.1.1 Functional class name

78 The class name section provides information necessary to identify and categorise a functional class. Every functional class has a unique name, consisting of a short name of three characters. The short name of the class is used in the specification of the short names of the families of that class.

7.1.1.2 Class introduction

79 The class introduction expresses the common intent or approach of those families to satisfy security objectives. The definition of functional classes does not reflect any formal taxonomy in the specification of the components.

80 The class introduction provides a figure describing the families in this class and the hierarchy of the components in each family, as explained in section 7.1.4.

7.1.2 Functional family structure

81 Figure 5 illustrates the functional family structure in diagrammatic form.

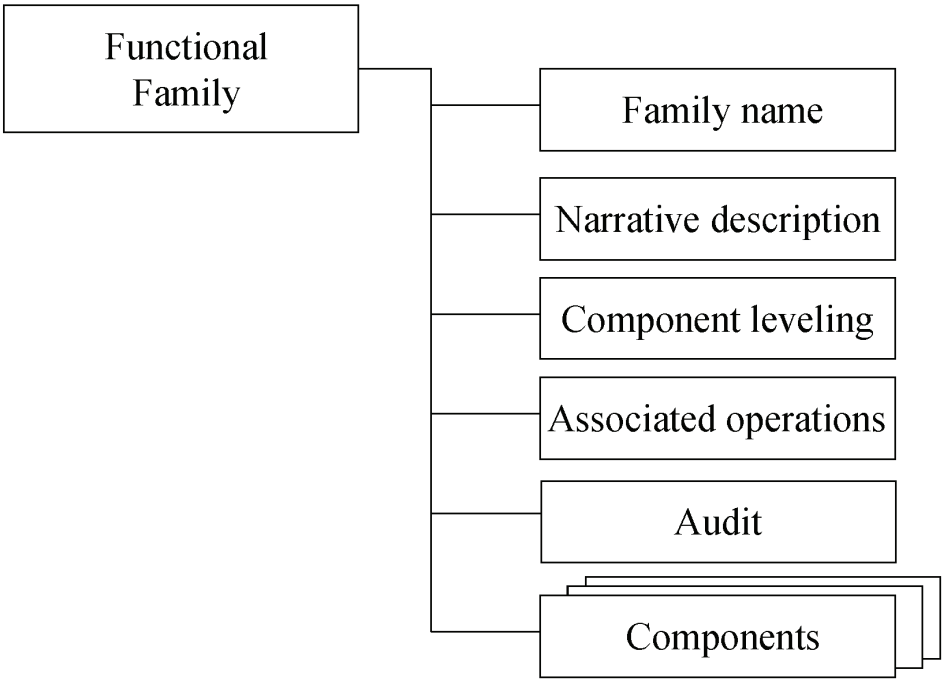


Figure 5 - Functional family structure

7.1.2.1 Family name

82 The family name section provides categorical and descriptive information necessary to identify and categorise a functional family. Every functional family has a unique name. The categorical information consists of a short name of seven characters, with the first three identical to the short name of the class followed by an underscore and the short name of the family as follows FXX_YYY. The unique short form of the family name provides the principal reference name for the components.

7.1.2.2 Narrative description

83 This section provides a narrative description of the family as a whole and, where applicable, each individual component. This narrative description is aimed at authors of PPs, STs and functional packages who wish to assess whether the family is relevant to their specific PP or ST.

7.1.2.3 Component levelling

84 Functional families contain one or more components, any one of which may be selected for inclusion in PPs, STs and functional packages. The goal of

this section is to provide information to users in selecting an appropriate functional component once the family has been identified as being necessary or useful.

85 The relationships between components within a functional family may be hierarchical. A component is hierarchical to another if the first component is more restrictive than the second, i.e. a TOE meeting the second requirement would also meet the first, but a TOE meeting the first requirement would not always meet the second requirement.

86 The descriptions of the families provide a graphical overview of the hierarchy of the components in a family.

7.1.2.4 Associated operations

87 The *associated operations* section for a given security functional component contains suggestions for operations (actions of subjects on objects) that may be useful in conjunction with that functional component. Where applicable the suggested object to apply the operations to has been listed as well. PP/ST authors may consider including these in their PP/ST by including them in their list of defined operations, and using Access control (FDP_ACC) components to specify which subject may do these operations.

88 PP/ST authors do not have to include these operations, nor do they have to justify not including these operations into their PP/ST. They may include other operations related to this specific security functional component in their PP/ST.

89 This Part 2 uses a particular notation for operations (natural language words separated by _ characters.). This notation is neither mandatory nor specifically recommended: any notation can be used as long as it is coherent and consistently used.

7.1.2.5 Audit

90 The audit section for a given security functional component contains suggestions for events that may usefully be audited. If the PP/ST contains a component of the FAU_GEN family, PP/ST authors may consider including these events (or others) in the definition of the Security audit data generation (FAU_GEN) SFR.

91 PP/ST authors do not have to include these events, nor do they have to justify not including these events into their PP/ST. They may include other events related to this specific security functional component in their PP/ST.

7.1.3 Functional component structure

92 Figure 6 illustrates the functional component structure.

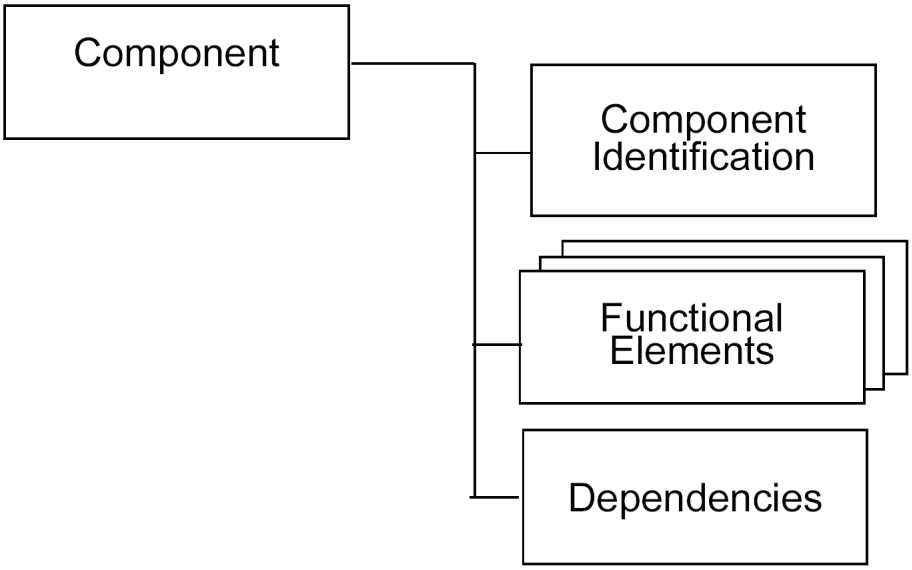


Figure 6 - Functional component structure

7.1.3.1 Component identification

93 The component identification section provides descriptive information necessary to identify, categorise, register and cross-reference a component. The following is provided as part of every functional component:

94 A *unique name*. The name reflects the purpose of the component.

95 A *short name*. A unique short form of the functional component name. This short name serves as the principal reference name for the categorisation, registration and cross-referencing of the component. This short name reflects the class and family to which the component belongs and the component number within the family.

96 A *hierarchical-to list*. A list of other components that this component is hierarchical to and for which this component can be used to satisfy dependencies to the listed components.

7.1.3.2 Functional elements

97 A set of elements is provided for each component. Each element is individually defined and is self-contained.

98 A functional element is a statement that if further divided would not yield a meaningful evaluation result. It is the smallest functional statement identified and recognised in the CC.

99 When deriving SFRs from a component (i.e. when building packages, PPs and/or STs), it is not permitted to select only one or more elements from a component. The complete set of elements of a component shall be used to create an SFR and the complete SFR shall be included in a PP, ST or package.

100 A unique short form of the functional element name is provided. For example the requirement name FIA_AFL.1.2 reads as follows: F - functional requirement, IA - class “Identification, Authentication and Binding”, _AFL - family “Authentication failures”, .1 - 1st component named “Authentication Failure Handling”, .2 - 2nd element of the component.

7.1.3.3 Dependencies

101 Each functional component provides a complete list of dependencies to other functional components. Some components may list “No dependencies”.

102 The components depended upon may in turn have dependencies on other components. The list provided in the components will be only direct dependencies. The indirect dependencies, that is the dependencies that result from the depended upon components may be found in Annex A of this part of the CC. In some cases, the dependency is optional in that a number of functional requirements are provided, where each one of them would be sufficient to satisfy the dependency. For more information on dependencies, see CC Part 1 Annex C.3.

7.1.4 Functional component catalogue

103 The grouping of the components in this part of the CC does not reflect any formal taxonomy.

104 This part of the CC contains classes of families and components, which are rough groupings on the basis of related function or purpose. At the start of each class is an informative diagram that indicates the taxonomy of each class, indicating the families in each class and the components in each family. The diagram is a useful indicator of the hierarchical relationship that may exist between components.

105 In the description of the functional components, a section identifies the dependencies between the component and any other components.

106 In each class a figure describing the family hierarchy similar to Figure 7, is provided.

107 In Figure 7 the first family, Family 1, contains three hierarchical components, where component 2 and component 3 may both be used to satisfy dependencies on component 1. Component 3 is hierarchical to component 2 and may also be used to satisfy dependencies on component 2.

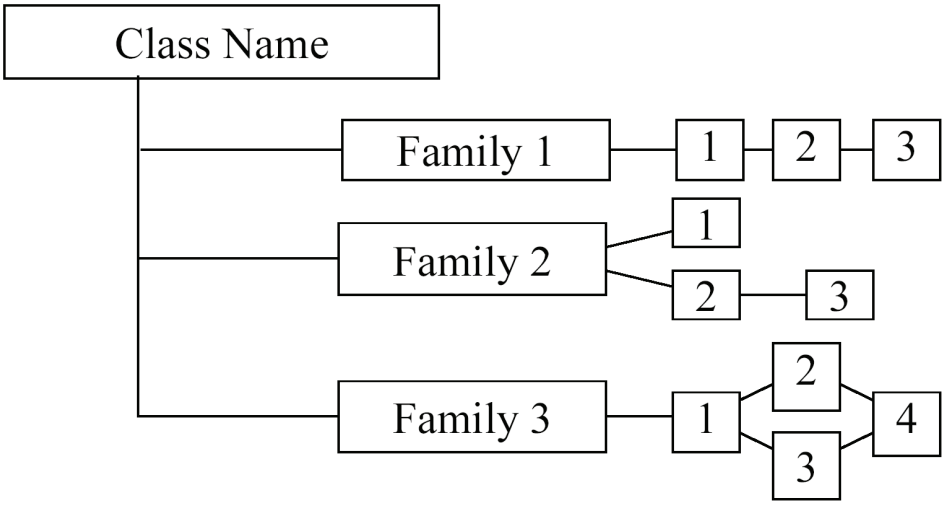


Figure 7 - Sample class decomposition diagram

108 In Family 2, there are three components not all of which are hierarchical. Components 1 and 2 are hierarchical to no other components. Component 3 is hierarchical to component 2, and may be used to satisfy dependencies on component 2, but not to satisfy dependencies on component 1.

109 In Family 3, components 2, 3, and 4 are hierarchical to component 1. Components 2 and 3 are both hierarchical to component 1, but non-comparable. Component 4 is hierarchical to both component 2 and component 3.

7.1.4.1 Component changes highlighting

110 The relationship between components within a family is highlighted using a bolding convention. This bolding convention calls for the bolding of all new requirements. For hierarchical components, (parts of) elements and/or dependencies are bolded when they are enhanced or modified beyond the requirements of the previous component. In addition, any new or enhanced permitted operations beyond the previous component are also highlighted using bold type.

8 Overview of functional classes

111 The paradigm described in Chapter 6 leads to a number of functional classes: groups of security functional components with a similar goal. The conceptual organisation of these classes is as follows:

8.1 Data Protection and Privacy (FDP)

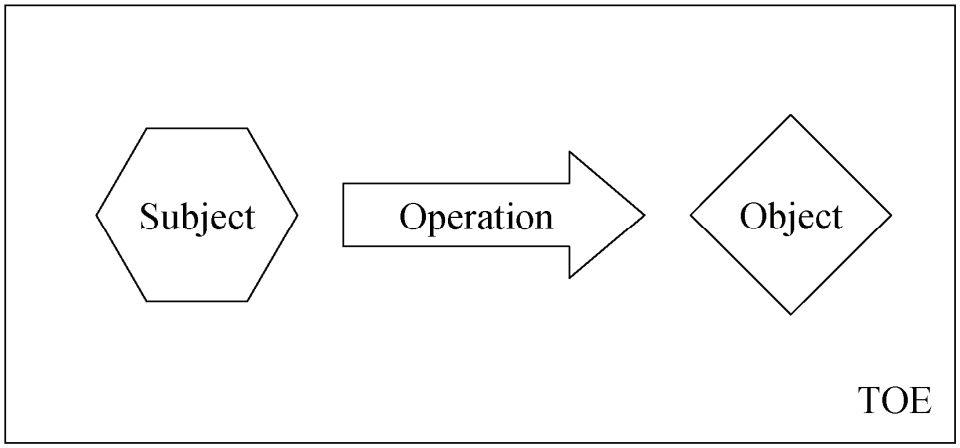


Figure 8 - Data Protection and Privacy

112 This class is intended to describe the internal security behaviour of the TOE. To meet this goal it defines components for the interaction between subjects and objects (operations). This class also defines criteria for security attributes: how they obtain an initial value and how they are subsequently modified.

8.2 Identification, authentication and binding (FIA)

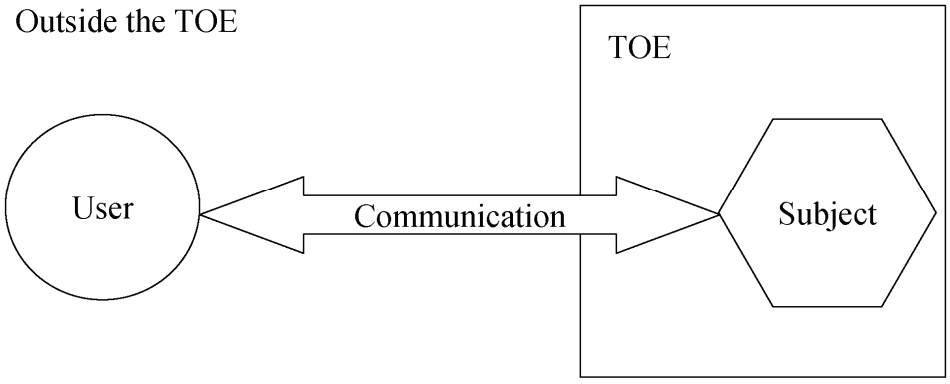


Figure 9 - Identification, Authentication and Binding

113 This class is intended to define how users may bind to subjects in the TOE (identification, authentication and other conditions), what the consequences of this binding are for the subject, and under which conditions the binding is dissolved.

8.3 Communication (FCO)

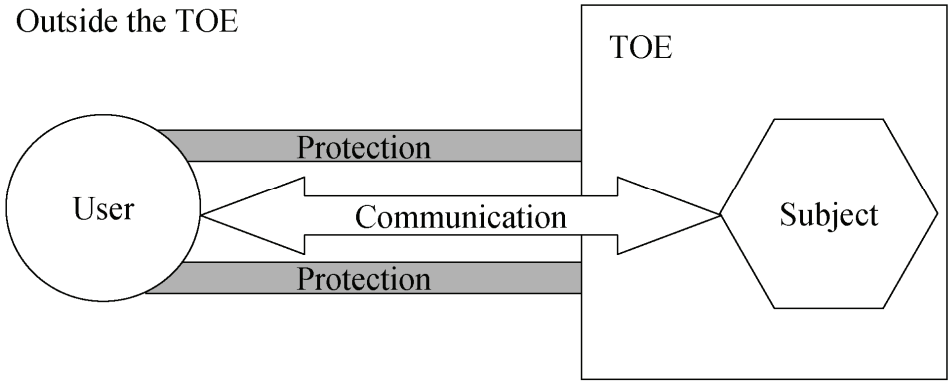


Figure 10 - Communication

114 Once a user has been bound to a subject he may communicate with that subject. This class defines components that address the protection of the communication (i.e. guaranteeing confidentiality, integrity and availability) between subjects and users bound to those subjects.

8.4 Audit (FAU)

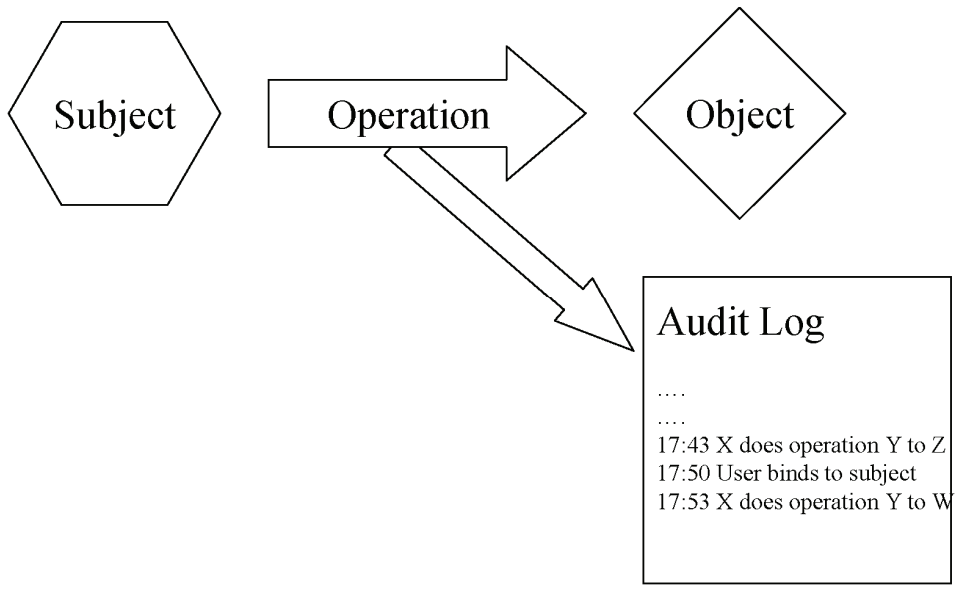


Figure 11 - Audit

115 This class defines components for logging events, automatically analysing them and acting on the results of this analysis. The purpose for this is twofold:

a) Detecting certain security-relevant events and responding to them as or after they occur;

b) Detecting certain security relevant events and tracing these back through subjects to particular users that are ultimately responsible for these events, so that these users may be held accountable for their actions.

8.5 Protection of the TSF (FPT)

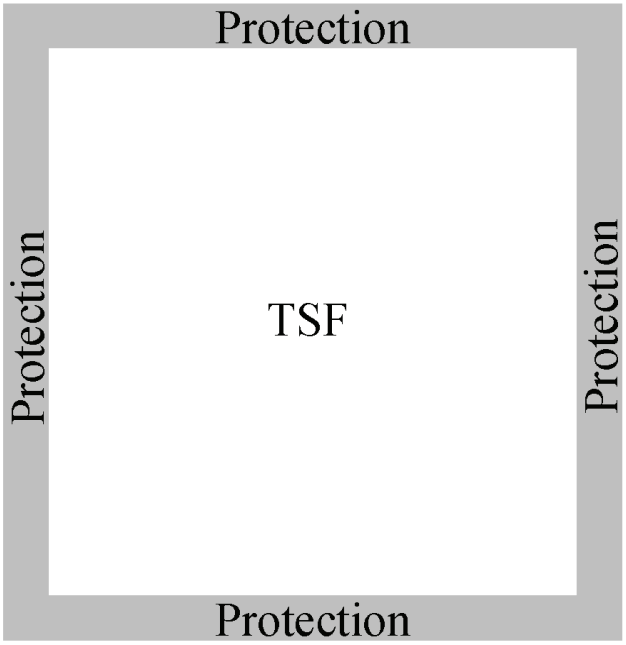


Figure 12 - Protection of the TSF

116 Other functional classes describe protection provided by the TSF, but assume that the TSF itself is physically protected, that TSF integrity is maintained, and that the TSF has an infinite amount of idealised resources at its disposal.

117 This class defines families for self-protection of the TSF. It includes components for:

a) testing, breakdown and recovery of the TSF;

b) tamper-evidence, tamper-detection, tamper-responsiveness and tamper-resistance of the TSF

c) prioritisation of resources and permanent removal of data from resource

8.6 Miscellaneous (FMI)

118 This class defines miscellaneous components that did not readily fit into another class but warranted no class of their own: random number generation, time-stamping and a requirement for allowing a choice between different sets of SFRs.

9 Class FDP: Data protection and privacy

119 This class contains families that can be used to specify requirements for the protection of data while it is inside the TOE. To this end it provides components to describe objects, subjects, operations, attributes and their interactions.

120 The FDP: Data protection and privacy class does not contain explicit components for traditional Mandatory Access Controls (MAC) or traditional Discretionary Access Controls (DAC); however, TSPs that require these constructs can be constructed using components from this class.

121 The class is structured as follows:

9.1 Operations

a) Access control (FDP_ACC) should be used for specifying rules for the performing of operations on objects by subjects, based on security attributes;

b) Rollback (FDP_ROL) should be used to specify rules for undoing these operations;

c) Unobservability (FDP_UNO) should be used to specify whether subjects are unable to observe other subjects doing operations;

d) Unlinkability (FDP_UNL) should be used to specify whether subjects are unable to link different subjects, objects and/or operations together in some way;

9.2 Security Attributes

a) Initialisation of security attributes (FDP_ISA) should be used to specify rules for the initialising of security attributes when new objects or subjects are created;

b) Management of security attributes (FDP_MSA) should be used to specify rules for the access to and modification of values of security attributes of existing subjects and objects.

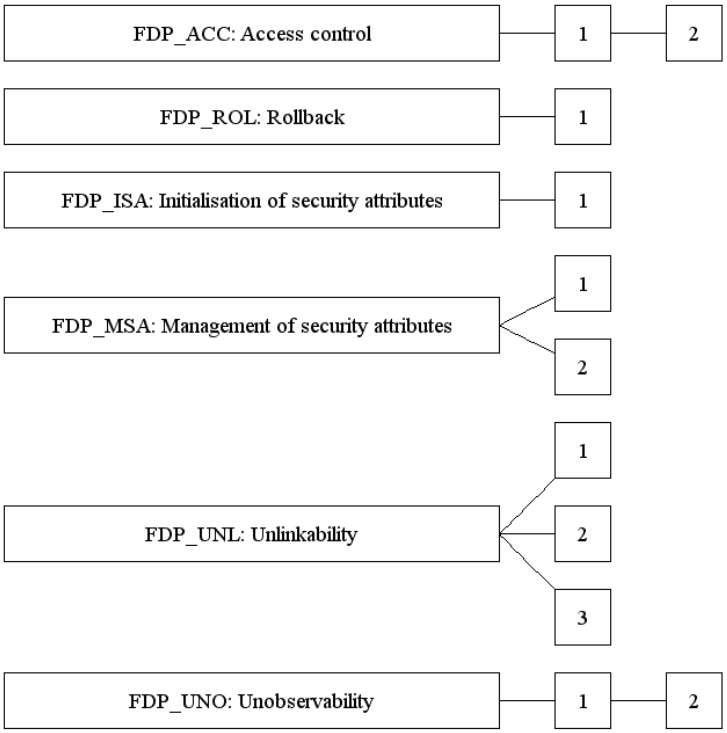


Figure 13 - FDP: Data protection and privacy class decomposition

9.3 Access control (FDP_ACC)

122 Access control consists of allowing or disallowing subjects to perform a specified operation on objects. The decision to allow or disallow is based solely on the security attributes of the involved subjects and objects.

123 The Access control (FDP_ACC) family specifies access control in terms of "operations". An operation is defined as a specific type of action of a subject on a specific object. It depends on the level of abstraction of the PP/ST author whether these operations are described as "read" and/or "write" operations, or as more detailed operations such as "update the database".

124 A typical access control rule would be: "any subject with owner=admin may execute the read, write, create and destroy operations on all objects of type=file", where owner and type are security attributes.

125 For the sake of readability it is often good to split the access control requirements over multiple iterations of Access control (FDP_ACC), e.g. one iteration dealing with data files and the other iteration dealing with emails. The PP/ST author may use the same subject, object or operation in multiple iterations of Access control (FDP_ACC), as long as it is clear and consistent.

126 It is not required to specify for every combination of object, subject or operation whether it is allowed or not allowed. However, if a combination is not specified, this means that both TOEs that do allow the operation and TOEs that do not allow the operation would meet this SFR.

127 An important aspect of access control is the ability to modify the security attributes that are involved in the access control decisions. This is done through components of the Management of security attributes (FDP_MSA) family.

128 The Access control (FDP_ACC) family consists of two components:

a) FDP_ACC.1 Access control: this may be used to specify relations between objects, subjects and operations based on security attributes.

b) FDP_ACC.2 Access control with automatic modification of security attributes: this may be used similarly to FDP_ACC.1 Access control, but with the added possibility of modifying the values of security attributes based on the success/failure of an operation. This allows specification of well-known security models such as: the Bell and LaPadula Security model BL and the Biba Integrity model BIBA;

129 Access control (FDP_ACC) has no specific components to specify functionalities such as two-person control, sequence rules for operations, or exclusion controls, but these may be specified by careful drafting of the access control rules in existing Access control (FDP_ACC) components.

Component levelling

Associated operations: FDP_ACC.1, FDP_ACC.2

130 None.

Audit: FDP_ACC.1, FDP_ACC.2

131 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful operations, possibly with the values of the security attributes used to make the access control decision

FDP_ACC.1 Access control

Hierarchical to: No other components.

Dependencies: FDP_ISA.1 Security attribute initialisation

FDP_ACC.1.1 **The TSF shall [selection: *allow, disallow*] an operation of a subject on an object [selection: *if, if and only if*] [assignment: *rules for operations, based on security attributes of the subjects and objects*].**

FDP_ACC.2 Access control with automatic modification of security attributes

Hierarchical to: FDP_ACC.1 Access control

Dependencies: FDP_ISA.1 Security attribute initialisation

FDP_ACC.2.1 The TSF shall [selection: *allow, disallow*] an operation of a subject on an object [selection: *if, if and only if*] [assignment: *rules for operations, based on security attributes of the subjects and objects*].

FDP_ACC.2.2 **The TSF shall change the security attributes of subjects and/or objects involved in operations as follows: [assignment: *rules for changing security attributes of subjects and/or objects involved in an operation, based on security attributes of the subjects and objects and whether the operation was allowed or disallowed*].**

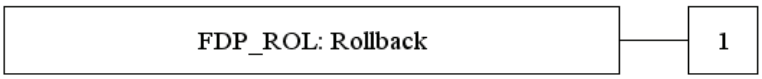
9.4 Rollback (FDP_ROL)

132 The rollback operation involves undoing the last operation or a series of operations, bounded by some limit, such as a period of time, and returning to a previous known state. Rollback provides the ability to undo the effects of an operation or series of operations while preserving the integrity of the TSF.

133 Special care should be taken when both Rollback (FDP_ROL) and Residual information protection (FPT_RIP) are present in the same PP or ST, as Residual information protection (FPT_RIP) may permanently delete information thereby making rollback impossible.

134 Rollback generally is bounded by certain limits: a database may store only so many backups, a disk may store only so many deleted files etc. Unless infinite rollback is supported, these limits should be specified in **FDP_ROL.1.2**.

Component levelling



Associated operations: FDP_ROL.1

135 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Change_Rollback_Boundarylimit: an operation that alters the boundary limit to which rollback may be performed. This operation should be applied to the TSF.

Audit: FDP_ROL.1

136 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful rollback attempts

b) The (types of) operations that were rolled back

FDP_ROL.1 Rollback

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FDP_ROL.1.1 **The TSF shall permit [assignment: *subject*] to rollback [assignment: *list of operations*] on [assignment: *list of objects*].**

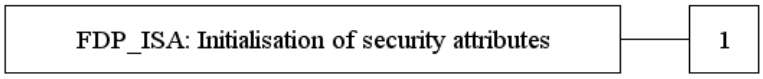
FDP_ROL.1.2 **The TSF shall permit operations to be rolled back within the [assignment: *boundary limit to which rollback may be performed*].**

9.5 Initialisation of security attributes (FDP_ISA)

137 Whenever new objects or subjects are created their security attributes shall be assigned initial values. This family allows specification of rules for these assignments.

138 This family is not to be confused with the User-subject binding (FIA_USB) family, which describes attribute changes in an existing subject when a user binds to it.

Component levelling



Associated operations: FDP_ISA.1

139 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Create_Subject: which creates a new subject;

b) Create_Object: which creates a new object.

Audit: FDP_ISA.1

140 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Creation of an object/subject together with the values that were assigned to the security attributes of that object/subject.

FDP_ISA.1 Security attribute initialisation

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FDP_ISA.1.1 **The TSF shall [selection: *use the following rules [assignment: rules] to assign an initial value , assign the value [assignment: value]*] to the security attribute [assignment: *security attribute*] whenever a [assignment: *object or subject*] is created.**

9.6 Management of security attributes (FDP_MSA)

141 This family should be used to describe the rules for the management of security attributes by subjects. This management might include capabilities for viewing and modifying of security attributes. This family consists of two components:

a) FDP_MSA.1 Management of security attributes, which should be used to specify the actions that subjects may take on security attributes and under which conditions these subjects are allowed to take those actions.

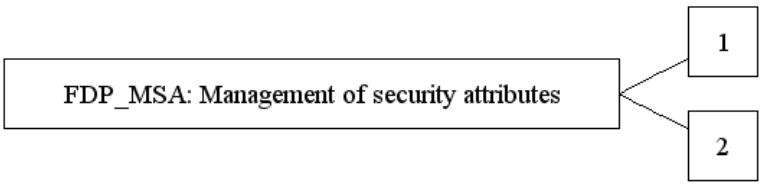
b) FDP_MSA.2 Automatic management of security attributes, which should be used to specify automatic changes in security attributes, such as "after ten minutes the subject loses its right to access the database" or "after the biometric data has been compared to the template it shall no longer be accessible for subjects".

142 It is possible to use FDP_MSA.1 Management of security attributes to specify situations where one subject may modify its own security attributes or two subjects may modify each others security attributes etc. As this may lead to TSPs that are difficult to understand and analyse, the PP/ST author should use these constructs only when needed.

143 Note that the value of security attributes can also be changed as the result of certain operations, if this is so specified by FDP_ACC.2 Access control with automatic modification of security attributes components. In this case, special care should be taken by the PP/ST author in ensuring that one SFR cannot be used to circumvent the other.

144 It is not required to specify for every combination of subject, security attribute and access type whether it is allowed or not allowed. However, if a combination is not specified, this means that both TOEs that do allow the access type and TOEs that do not allow the access type would meet this SFR.

Component levelling



Associated operations: FDP_MSA.1, FDP_MSA.2

145 None.

Audit: FDP_MSA.1

146 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful attempts to access security attributes, possibly with the values of the subject security attributes used to make the access decision;

b) The new values of security attributes;

c) The old values of security attributes.

Audit: FDP_MSA.2

147 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The new values of security attributes;

b) The old values of security attributes.

FDP_MSA.1 Management of security attributes

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FDP_MSA.1.1 **The TSF determine if a subject is allowed to [selection: *query, modify, delete, [assignment: other types of access] , [assignment: security attribute]*] or not, as follows: [assignment: *rules, based on the security attribute being accessed and the security attributes of the subject*].**

FDP_MSA.2 Automatic management of security attributes

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FDP_MSA.2.1 **If [assignment: *conditions*] are met, the TSF shall set the value of [assignment: *security attribute*] of [assignment: *subject or object*] according to [assignment: *rules*].**

9.7 Unlinkability (FDP_UNL)

148 This family ensures that subjects may perform several operations without other subjects being able to link these uses together, as the resulting aggregation of information may lead to an unwanted or unauthorised disclosure.

149 The requirements for unlinkability may be used to protect the identity of users bound to subjects against the use of profiling of the operations performed by those subjects. For example, when a telephone smart card is employed with a unique number, the telephone company may determine the behaviour of the user of this telephone card. When a telephone profile of the users is known, the card may be linked to a specific user. Hiding the relationship between different operation prevent this kind of information gathering.

150 Another use for unlinkability is when a series of phone calls made by an anonymous customer to different companies, where the linking together of the phone calls and the resulting combination of the companies' identities may disclose the identity of the customer.

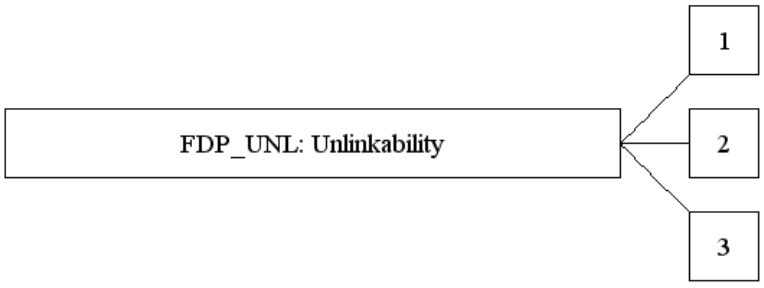
151 This family consists of three components:

a) FDP_UNL.1 Unlinkability of operations, that should be used to prevent subjects linking several operations together;

b) FDP_UNL.2 Unlinkability of subjects, that should be used to prevent subjects linking several subjects together;

c) FDP_UNL.3 Unlinkability of objects, that should be used to prevent subjects linking several objects together.

Component levelling



Associated operations: FDP_UNL.1, FDP_UNL.2, FDP_UNL.3

152 None.

Audit: FDP_UNL.1, FDP_UNL.2, FDP_UNL.3

153 There are no auditable events foreseen.

FDP_UNL.1 Unlinkability of operations

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_UNL.1.1 **The TSF shall ensure that [assignment: *subjects*] are unable to determine whether [assignment: *list of operations*] [selection: *were performed by the same subject, are related as follows [assignment: list of relations]*].**

FDP_UNL.2 Unlinkability of subjects

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_UNL.2.1 **The TSF shall ensure that [assignment: *subjects*] are unable to determine whether [assignment: *list of subjects*] [selection: *are related to the same object, are related by the same operation, are related as follows [assignment: list of relations]*].**

FDP_UNL.3 Unlinkability of objects

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_UNL.3.1 **The TSF shall ensure that [assignment: *subjects*] are unable to determine whether [assignment: *list of objects*] [selection: *are related to the same object, are related to the same subject, are related by the same operation , are related as follows [assignment: list of relations]*].**

9.8 Unobservability (FDP_UNO)

154 This family may be used to specify that a subject may perform an operation without other subjects being able to observe that the operation is being performed. This includes all side-effects of the operation, such as use of certain resources, changes in objects resulting from the operation, etc.

155 One of the uses of unobservability is to prevent a subject indirectly transferring information to another subject through the use of operations. An example of this is: Subject 1 chooses each second to either doing a lot of operations, or doing nothing. If Subject 2 may observe this or any side-effect of it, such as the TOE slowing down, the Subject 1 may signal Subject 2 at a rate of 1 bit/sec.

156 Note that this family is aimed at subjects observing subjects doing operations in the TOE. The related family Unobservability of export (FCO_UNE) is intended to deal with users observing users communicating with the TOE.

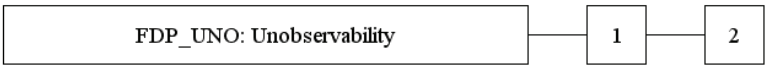
157 The obtaining of information (**FDP_UNO.1.1**) or act of observation (**FDP_UNO.2.1**) is not restricted to direct observation of a subject, but could also take indirect forms, such as examining the use of processing, storage or communication resources that may be used by the subject.

158 The Unobservability (FDP_UNO) family consists of two components:

a) FDP_UNO.1 Limited unobservability which should be used to specify that certain subjects may gain only limited information from observing other subjects doing certain operations;

b) FDP_UNO.2 Full unobservability which should be used to specify that certain subjects shall gain no information from observing other subjects doing certain operations.

Component levelling



Associated operations: FDP_UNO.1, FDP_UNO.2

159 None.

Audit: FDP_UNO.1, FDP_UNO.2

160 There are no auditable events foreseen.

FDP_UNO.1 Limited unobservability

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_UNO.1.1 **The TSF shall ensure that [assignment: *subjects*] are unable to obtain more than [assignment: *measure for information*] from observing [assignment: *list of operations*] on [assignment: *list of objects*] by [assignment: *list of subjects*].**

FDP_UNO.2 Full unobservability

Hierarchical to: FDP_UNO.1 Limited unobservability

Dependencies: No dependencies.

FDP_UNO.2.1 The TSF shall ensure that [assignment: ***list of subjects***] are unable to **observe** [assignment: *list of operations*] on [assignment: *list of objects*] by [assignment: *list of subjects*].

10 Class FIA: Identification, Authentication and Binding

161 The TOE communicates with the outside world by allowing users (active entities outside the TOE) to bind with subjects (active entities inside the TOE).

162 The families in the FIA: Identification, Authentication and Binding class deal with registering and unregistering new users, determining and verifying the identity of registered users, binding and unbinding users to/from subjects, and determining the correct capabilities of subjects that are bound to and unbound from users.

163 The FIA: Identification, Authentication and Binding class contains a substantial number of families. These families are best understood as parts of an overall process, rather than as individual families. This process is described below.

10.1 Before a user first binds to a subject:

a) User registration (FIA_URE). Users may need to register. As part of this registering they:

   1) may receive user security properties (e.g. identifiers, access rights) and,

   2) may supply and/or receive authentication data (e.g. passwords, biometric information.

b) Quality of Authentication Data (FIA_QAD). This authentication data may need to have some quality to ensure that e.g. a password shall not be trivially guessed.

10.2 Every time a user attempts to bind to a subject:

a) User identification (FIA_UID). When a user attempts to bind to a subject, identification may be needed or bindings may be anonymous.

b) User authentication (FIA_UAU). For some bindings authentication may be needed. In this case the TSF may do the authentication itself, or the TSF may obtain the authentication status elsewhere (e.g. from the operating system for an application TOE). Several additional options exist for authentication:

   1) The authentication may need to be unforgeable, so that one user is unable to steal or lend it from another. A typical implementation of this component is the use of biometric mechanisms;

2) The authentication may need to be single-use, so that eavesdropping on the authentication is pointless. A typical implementation of this component is the use of a list of one-time passwords;

3) In some cases a user may need to re-authenticate, for instance if the user attempts to perform a security-critical action or when the user has been inactive for a long time;

4) The authentication may need to be protected, so that a person observing the user doing authentication is unable to obtain the authentication data. A typical implementation of this component consists of displaying only "*" when a human user types in a password or PIN.

c) Authentication failures (FIA_AFL). The authentication may fail (repeatedly) and based on this failure the TSF may perform actions, such as warning the administrator or disabling the account;

d) TSF binding rules (FIA_TBR). Even if the authentication is successful, the TSF may decide to not allow a user to bind to a subject anyway, for instance, because the user tries to bind outside normal operating hours.

10.3 Every time a user is bound to a subject:

a) User-subject binding (FIA_USB). Depending on the specifics of the binding, some or all of the user security properties will now be transformed to security attributes of the subject being bound to. This will allow the subject to properly represent the user inside the TOE.

b) Subject/TSF authentication (FIA_SUA). The subject (or the TSF) may have to authenticate itself to the user as well. This will allow the user to ensure that the subject (or the TSF) is not being impersonated.

c) TSF Information (FIA_TIN). The TSF may send a warning banner concerning proper use of the TSF, or the access history of the TSF or a specific subject to a user binding to a subject.

d) Lock-out of bindings (FIA_LOB). The binding may be locked-out temporarily freezing the binding until the binding is unlocked.

e) Termination of bindings (FIA_TOB). The binding may be terminated thereby ending the binding and possibly resetting the security attributes of the subject that were set in User-subject binding (FIA_USB).

When bound, the user may communicate with the subject and order the subject to perform operations. Components to specify this may be found in the FCO and FDP classes.

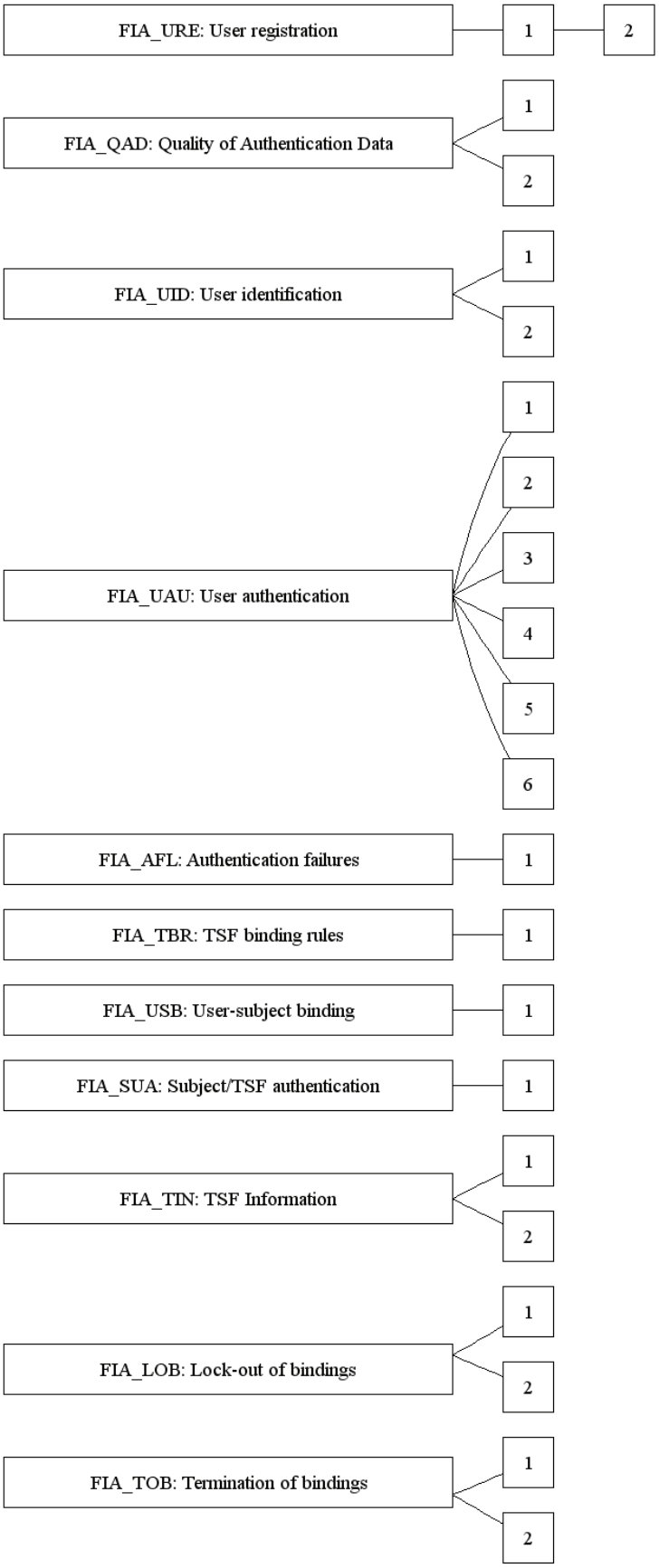


Figure 14 - FIA: Identification, Authentication and Binding class decomposition

10.4 User registration (FIA_URE)

164 This family allows new users to register themselves. As part of this registration process the TSF may generate and store user security properties of that user. The TSF may also receive or generate authentication data for that user and store this.

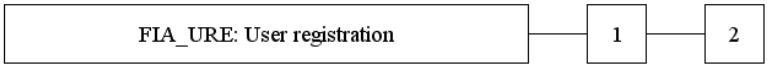
165 CC Part 2 does not define components to remove/modify the registration of a user. As the user security properties and authentication data are stored in objects, this should be modelled by defining operations (preferably using the Associated Operations defined for this component). Access to these operations should then be defined using components from the Access control (FDP_ACC) family.

166 This family consists of two components:

a) FIA_URE.1 User registration without storage of authentication data, which should be used when only user security properties are stored for registering users;

b) FIA_URE.2 User registration with storage of authentication data, which should be used when both user security properties and authentication data is stored for registering users.

Component levelling



Associated operations: FIA_URE.1

167 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Remove_User_Security_Properties: an operation that removes user security properties. This operation should be applied to the object used to store the user security properties in.

b) Modify_User_Security_Properties: an operation that modifies user security properties. This operation should be applied to the object used to store the user security properties in.

Associated operations: FIA_URE.2

168 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Remove_User_Security_Properties: an operation that removes user security properties. This operation should be applied to the object used to store the user security properties in.

b) Modify_User_Security_Properties: an operation that modifies user security properties. This operation should be applied to the object used to store the user security properties in.

c) Remove_Authentication_Data: an operation that removes authentication data. This operation should be applied to the object used to store authentication data in.

d) Modify_Authentication_Data: an operation that modifies the authentication data of a user (e.g. changing a password). This operation should be applied to the object used to store the user security properties in.

Audit: FIA_URE.1

169 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Registration of a user, possibly with (a subset of) the associated user security properties.

b) Any successful or unsuccessful operation on the object used to store user security properties.

Audit: FIA_URE.2

170 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Registration of a user, possibly with (a subset of) the associated user security properties,

b) Any successful or unsuccessful operation on the object used to store user security properties.

c) Any successful or unsuccessful operation on the object used to store user security properties.

FIA_URE.1 User registration without storage of authentication data

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FIA_URE.1.1 **The TSF shall be able to register new users.**

FIA_URE.1.2 **The TSF shall [selection: *obtain values for [assignment: user security properties] from the registering user , provide values for [assignment: user security properties] as follows: [assignment: rules for deriving security properties for the registering user]*].**

FIA_URE.1.3 **The TSF shall store these user security properties in [assignment: *object*].**

FIA_URE.2 User registration with storage of authentication data

Hierarchical to: FIA_URE.1 User registration without storage of authentication data

Dependencies: FDP_ACC.1 Access control

FIA_URE.2.1 The TSF shall be able to register new users.

FIA_URE.2.2 The TSF shall [selection: *obtain values for [assignment: user security properties] from the registering user , provide values for [assignment: user security properties] as follows: [assignment: rules for deriving security properties for the registering user]*].

FIA_URE.2.3 The TSF shall store these user security properties in [assignment: *object*].

FIA_URE.2.4 **The TSF shall [selection: *receive authentication data from the registering user, provide authentication data to the registering user, [assignment: other method to establish authentication data between the registering user and the TSF]*].**

FIA_URE.2.5 **The TSF shall store this authentication data in [assignment: *object*].**

10.5 Quality of Authentication Data (FIA_QAD)

171 Even the most sophisticated authentication mechanisms may be broken if the authentication data that is used is of low quality, e.g. a password consisting of "Enter". With this family, the PP/ST author may ensure that the authentication data being used is of sufficient quality;

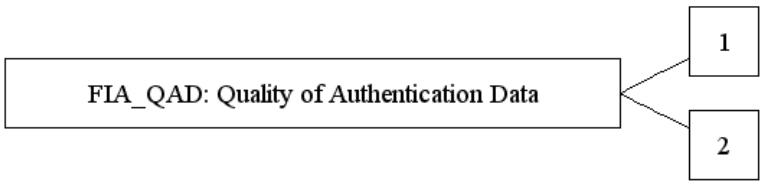
172 Note that the quality metric that is being assigned shall be objective and testable. A completion of "hard to guess" is not objective or testable but "a password length of 8 randomly selected ASCII characters" is.

173 This family consists of two components:

a) FIA_QAD.1 Verification of quality of authentication data, which should be used to specify that user-supplied authentication data is checked for quality.

b) FIA_QAD.2 TSF generation of authentication data, which should be used to specify that the TSF generates the authentication data, and that this authentication data has sufficient quality.

Component levelling



Associated operations: FIA_QAD.1, FIA_QAD.2

174 None.

Audit: FIA_QAD.1

175 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Acceptance or rejection of user-supplied authentication data.

Audit: FIA_QAD.2

176 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Generation of authentication data.

FIA_QAD.1 Verification of quality of authentication data

Hierarchical to: No other components.

Dependencies: FIA_URE.2 User registration with storage of authentication data
FIA_USB.1 User-subject binding

FIA_QAD.1.1 **The TSF shall ensure that authentication data needed to bind to [assignment: *subject*] meets [assignment: *quality metric*].**

FIA_QAD.2 TSF generation of authentication data

Hierarchical to: No other components.

Dependencies: FIA_URE.2 User registration with storage of authentication data
FIA_USB.1 User-subject binding

FIA_QAD.2.1 **The TSF shall be able to generate authentication data that meets [assignment: *quality metric*].**

FIA_QAD.2.2 **The TSF shall enforce the use of this authentication data for authentication related to binding to [assignment: *subject*].**

10.6 User identification (FIA_UID)

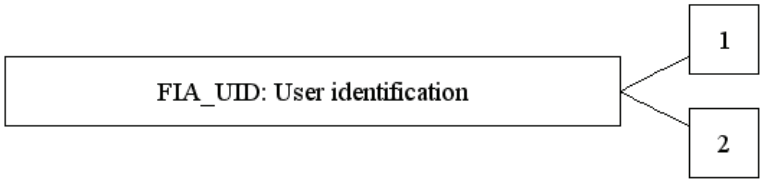
177 This family provides components that may be used to specify identification of users. Two types of identification exist:

a) anonymous users are allowed to bind to a subject (FIA_UID.1 Anonymous users);

b) users shall be identified before they may bind to a subject (FIA_UID.2 User identification).

178 If a subject is not covered by any iteration of a component of this family, users are not allowed to bind to that subject.

Component levelling



Associated operations: FIA_UID.1, FIA_UID.2

179 None.

Audit: FIA_UID.1

180 There are no auditable events foreseen.

Audit: FIA_UID.2

181 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The claimed identity of a user.

FIA_UID.1 Anonymous users

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FDP_UID.1.1 **The TSF shall allow users to bind to [assignment: *subject*] without identifying themselves.**

FIA_UID.2 User identification

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_UID.2.1 **The TSF shall identify a user before the user can bind to [assignment: *subject*].**

10.7 User authentication (FIA_UAU)

182 This family provides components that may be used to specify that users shall be authenticated before they may bind to a subject. This authentication may be done either by the TSF (FIA_UAU.1 User authentication by TSF), or the TSF may obtain the authentication status from a third party outside the TOE such as the operating system for an application TOE (FIA_UAU.2 User authentication by third party).

183 In addition, this family contains components that allow specification of several additional authentication options:

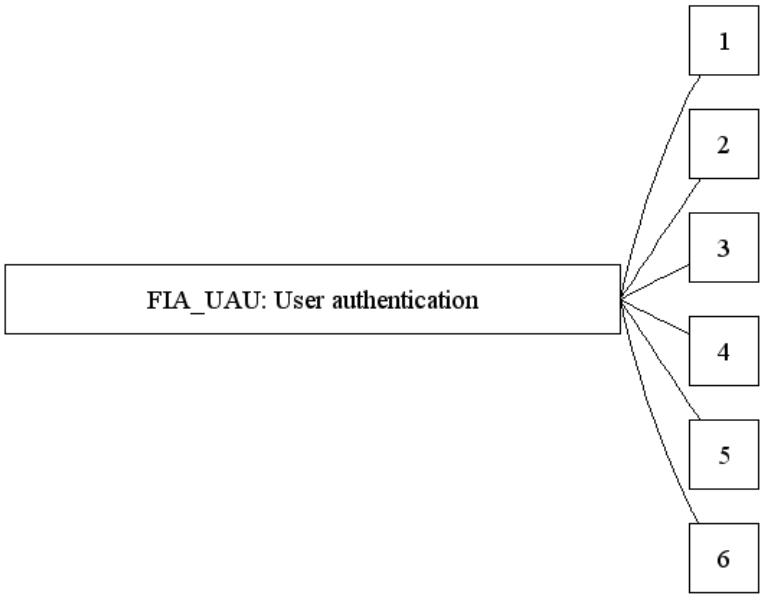
a) FIA_UAU.3 Unforgeable authentication, this component should be used to specify unforgeable authentication methods that prevent users from stealing or lending authentication data from each other. A typical implementation of this component is the use of biometric mechanisms;

b) FIA_UAU.4 Single-use authentication, the authentication may need to use particular authentication data only one time, so that eavesdropping on the authentication is pointless. A typical implementation of this component is the use of a list of one-time passwords;

c) FIA_UAU.5 Re-authentication, which can describe cases where a user may need to re-authenticate, for instance if the user attempts to perform a security-critical action or when the user has been inactive for a long time;

d) FIA_UAU.6 Limited authentication feedback, which can be used to specify that the TSF limits the feedback it gives to the user in the authentication process, so that other users observing the user doing authentication are unable to obtain the authentication data from this feedback. A typical implementation of this component consists of displaying only "*" when a human user types in a password or PIN.

Component levelling



Associated operations: FIA_UAU.1, FIA_UAU.2, FIA_UAU.3, FIA_UAU.4, FIA_UAU.5, FIA_UAU.6

184 None.

Audit: FIA_UAU.1

185 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful attempts at authentication.

Audit: FIA_UAU.2

186 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each request for authentication from the third party and its result.

Audit: FIA_UAU.3

187 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Detection of forged or copied authentication data;

b) The actions undertaken upon detection.

Audit: FIA_UAU.4

188 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Attempts to re-use authentication data.

Audit: FIA_UAU.5

189 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful re-authentication attempts.

Audit: FIA_UAU.6

190 There are no auditable events foreseen.

FIA_UAU.1 User authentication by TSF

Hierarchical to: No other components.

Dependencies: FIA_UID.2 User identification
FIA_URE.2 User registration with storage of authentication data

FIA_UAU.1.1 The TSF shall authenticate a user before the user can bind to [assignment: *subject*].

FIA_UAU.2 User authentication by third party

Hierarchical to: No other components.

Dependencies: FIA_UID.2 User identification

FIA_UAU.2.1 The TSF shall verify that a user has been authenticated by [assignment: *other user*] before the user can bind to [assignment: *subject*].

FIA_UAU.3 Unforgeable authentication

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 User authentication by TSF

FIA_UAU.3.1 The TSF shall [selection: *detect attempted, detect successful, prevent successful*] use of forged authentication data.

FIA_UAU.3.2 The TSF shall [selection: *detect attempted, detect successful, prevent successful*] use of copied authentication data.

FIA_UAU.3.3 If the TSF detects the use of [selection: *forged, copied*] authentication data, the TSF shall [assignment: *list of actions*].

FIA_UAU.4 Single-use authentication

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 User authentication by TSF

FIA_UAU.4.1 **The TSF shall prevent successful reuse of authentication data.**

FIA_UAU.5 Re-authentication

Hierarchical to: No other components.

Dependencies: [FIA_UAU.1 User authentication by TSF, or
FIA_UAU.2 User authentication by third party]

FIA_UAU.5.1 **The TSF shall re-authenticate the user if [assignment: *list of conditions under which re-authentication is required*].**

FIA_UAU.5.2 **When this re-authentication fails, the TSF shall [selection: *unbind the user from [assignment: subject] , perform [assignment: action]*].**

FIA_UAU.6 Limited authentication feedback

Hierarchical to: No other components.

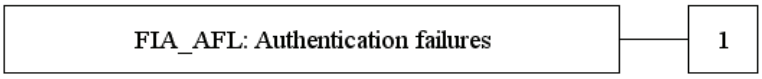
Dependencies: FIA_UAU.1 User authentication by TSF

FIA_UAU.6.1 **The TSF shall provide only [assignment: *list of feedback*] to the user while the authentication is in progress.**

10.8 Authentication failures (FIA_AFL)

191 This family contains components for defining values for some number of unsuccessful authentication attempts and TSF actions in case this number is exceeded. Examples of TSF actions are warning the administrator or disabling the account

Component levelling



Associated operations: FIA_AFL.1

192 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Change_Authentication_Threshold: an operation that alters the number of authentication failures to be met or surpassed. This operation should be applied to the TSF.

Audit: FIA_AFL.1

193 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Meeting or surpassing the threshold;

b) The actions undertaken upon meeting or surpassing the threshold.

FIA_AFL.1 Authentication failure handling

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 User authentication by TSF

FIA_AFL.1.1 **The TSF shall detect when [assignment: *positive integer*] unsuccessful authentication attempts occur related to [selection: *the same user, the same subject, [assignment: other common property of the unsuccessful authentication attempts]*].**

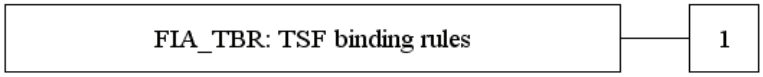
FIA_AFL.1.2 **When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [assignment: *list of actions*].**

10.9 TSF binding rules (FIA_TBR)

194 This family defines requirements to deny a user permission to bind to a subject based on other reasons than a failed authentication. These reasons could include the time that a user attempts to bind, or the interface that the user is attempting to bind through.

195 These reasons could be combined with user security properties of the user, e.g. to specify that a user with clearance Secret may bind between 20:00 and 06:00, but a user with clearance Confidential may not.

Component levelling



Associated operations: FIA_TBR.1

196 None.

Audit: FIA_TBR.1

197 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful bindings;

b) The specific values for parameters and user security properties that were used to allow/deny a binding.

FIA_TBR.1 TSF binding rules

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

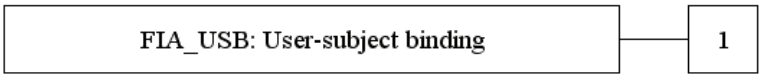
FIA_TBR.1.1 **The TSF shall deny a user binding to [assignment: *subject*] if [assignment: *rules based on one or more of user security properties, location of access, time of access, number of existing bindings of that user, other parameters*].**

10.10 User-subject binding (FIA_USB)

198 This family defines requirements to bind a subject to a user. For some bindings, there are no changes in the subject security attributes, it is simply now bound to a particular user. For some other bindings the security attributes of the subject are changed. This is done by transforming some of the user security properties into new values for the security attributes of that subject. This will allow the subject to properly represent the user inside the TSF.

199 The user security properties are typically determined at user registration (see the User registration (FIA_URE) family) and stored in an object on the TSF.

Component levelling



Associated operations: FIA_USB.1

200 None.

Audit: FIA_USB.1

201 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Some or all of the values of the resulting security attributes of the subject.

FIA_USB.1 User-subject binding

Hierarchical to: No other components.

Dependencies: No dependencies.

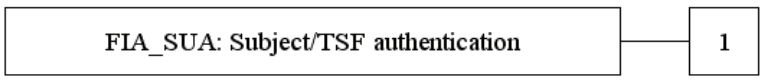
FIA_USB.1.1 **Upon binding a user to [assignment: *subject*] [selection: *the security attributes of the subject shall remain unchanged, the TSF shall change the values of security attributes of that subject as follows: [assignment: rules on how new values security attributes of that subject are determined from the user security properties of the user]*].**

10.11 Subject/TSF authentication (FIA_SUA)

202 This family defines requirements for a subject (or the TSF) to authenticate itself to a user. This will allow the user to ensure that the subject (or the TSF) is not being impersonated.

203 The subject/TSF authentication may take place before, after or during user authentication in User authentication (FIA_UAU).

Component levelling



Associated operations: FIA_SUA.1

204 None.

Audit: FIA_SUA.1

205 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each authentication of the subject/TSF.

FIA_SUA.1 TSF authentication

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

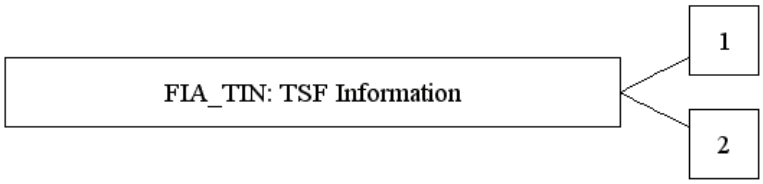
FIA_SUA.1.1 **[selection: *Before, As, After*] a user binds to [assignment: *subject*] the [selection: *subject, TSF*] shall authenticate itself to that user.**

10.12 TSF Information (FIA_TIN)

206 This family defines requirements to display a configurable advisory warning to users regarding the appropriate use of the TOE (FIA_TIN.1 Advisory warning message) and requirements to display the access history of that user to that user (FIA_TIN.2 TOE access history).

207 While these requirements do not provide additional technical security, FIA_TIN.1 Advisory warning message may be required for legal reasons, while FIA_TIN.2 TOE access history may allow a user to detect suspicious activity.

Component levelling



Associated operations: FIA_TIN.1

208 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Change_Advisory_Message: an operation that alters the advisory message. This operation should be applied to the TSF.

Associated operations: FIA_TIN.2

209 None.

Audit: FIA_TIN.1, FIA_TIN.2

210 There are no auditable events foreseen.

FIA_TIN.1 Advisory warning message

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_TIN.1.1 **Before allowing a user to bind to [assignment: *subject*], the TSF shall send an advisory warning message regarding unauthorised use of the TOE to the user.**

FIA_TIN.2 TOE access history

Hierarchical to: No other components.

Dependencies: FIA_UID.2 User identification

FIA_TIN.2.1 **Upon successful binding to [assignment: *subject*], the TSF shall send [selection: *the [selection: date, time, method, location] of the last successful binding to that subject by that user , the [selection: date, time, method, location] of the last unsuccessful attempt to bind to that subject by that user , the number of unsuccessful attempts to bind to that subject by that user since the last successful binding attempt by that user, [assignment: other message]*] to the user.**

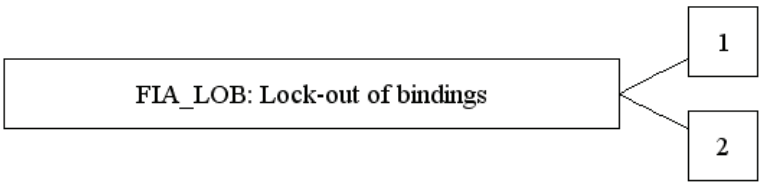
10.13 Lock-out of bindings (FIA_LOB)

211 This family defines requirements for the lock-out of bindings: the bindings are temporarily made inactive until specific conditions are met. While the binding is locked-out, users are still bound to the subjects, but the users shall be unable to communicate with the subjects they are bound to except to unlock the binding.

212 A typical implementation of a locked-out binding in an OS type TOE is starting a screen-saver, and disabling all input (mouse, keyboard) except the input necessary to re-authenticate the user.

213 Two different components are defined: FIA_LOB.1 TSF-initiated locking-out for when the TSF initiates the lock-out, and FIA_LOB.2 User-initiated locking out for when the user initiates the lock-out.

Component levelling



Associated operations: FIA_LOB.1, FIA_LOB.2

214 None.

Audit: FIA_LOB.1, FIA_LOB.2

215 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Lock-out of a binding;

b) Successful/unsuccessful unlocking of a binding.

FIA_LOB.1 TSF-initiated locking-out

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_LOB.1.1 **The TSF shall lock-out a binding to [assignment: *subject*] after [selection: *[assignment: time interval of user inactivity] , [assignment: other event(s)]*] by [selection: *clearing or overwriting display and/or communication devices, disabling any communication from/to the user bound to that subject except that necessary to unlock the binding, [assignment: other actions]*].**

FIA_LOB.1.2 **The TSF shall unlock the binding after [assignment: *event(s)*].**

FIA_LOB.2 User-initiated locking out

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_LOB.2.1 **The TSF shall allow a user to lock-out a binding of that user to [assignment: *subject*] by [selection: *clearing or overwriting display and/or communication devices, disabling any communication from/to the user bound to that subject except that necessary to unlock the binding, [assignment: other actions]*].**

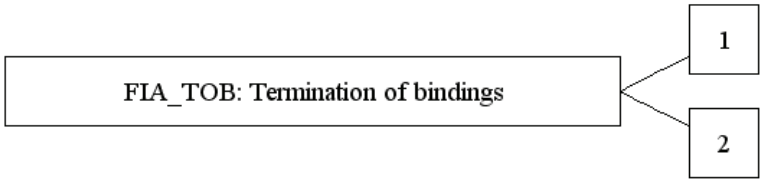
FIA_LOB.2.2 **The TSF shall unlock the binding after [assignment: *event(s)*].**

10.14 Termination of bindings (FIA_TOB)

216 This family defines requirements for the termination of bindings: the bindings are permanently inactivated and the subject that was bound to may have its security attributes reset.

217 Two different components are defined: FIA_TOB.1 TSF-initiated termination of binding for when the TSF initiates the termination and FIA_TOB.2 User-initiated termination of binding for when the user initiates the termination.

Component levelling



Associated operations: FIA_TOB.1, FIA_TOB.2

218 None.

Audit: FIA_TOB.1, FIA_TOB.2

219 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Termination of a binding

FIA_TOB.1 TSF-initiated termination of binding

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_TOB.1.1 **The TSF shall terminate a binding to [assignment: *subject*] after [selection: *completion of [assignment: operation], [assignment: time interval of user inactivity], [assignment: other condition]*].**

FIA_TOB.1.2 **The TSF shall [selection: *leave the security attributes of the subject unchanged, terminate the subject, set the security attributes of the subject to [assignment: rules for setting the security attributes of the subject]*].**

FIA_TOB.2 User-initiated termination of binding

Hierarchical to: No other components.

Dependencies: FIA_USB.1 User-subject binding

FIA_TOB.2.1 **The TSF shall allow a user to terminate a binding to [assignment: *subject*].**

FIA_TOB.2.2 **The TSF shall [selection:** ***leave the security attributes of the subject unchanged, terminate the subject, set the security attributes of the subject to [assignment: rules for setting the security attributes of the subject]*** **].**

11 Class FCO: Communication

220 With the FDP class, the internal working of the TOE can be modelled in terms of interactions between subjects ("internal processes in the TOE") and objects. With the FIA class, the binding process ("association" of a user (external entity) with a subject ("internal process") can be modelled. The FCO class is intended to model the communication and data exchange between a subject and a user bound to that subject.

221 If a user transfers data to a subject, this is called import of data. Examples of import include: restoring a backup, accepting an IP-packet from the WAN and accepting keyboard input from a human user.

222 If a subject transfers data to a user this is called export of data. Examples of export include: making a backup, sending an IP-packet to the LAN and providing output on a display for a human user.

223 Note that import and export are not considered to be operations in the CC, as they are subjects interacting with users and not subjects performing operations on objects. Also note that import and export do not transfer objects but data, as objects do not exist outside the TOE in the CC.

224 In many cases, the data exchange between subjects and users need to be limited, and in many cases, this data exchange between subjects and users needs to be protected against modification, disclosure and loss of use. The families in the FCO class are intended to allow a PP/ST writer to specify this protection.

225 Conceptually these families may be divided into two major groups, with a large degree of symmetry between the groups:

a) Export families: these families deal with protecting the communication from the subject to the user

b) Import families: these families deal with protecting the communication from the user to the subject.

226 Each of these groups is described in more detail below:

11.1 Export families

a) Export to outside TSF control (FCO_ETC) This family may be used to limit a subject from exporting all data to a user that the subject has access to. In addition, this family may be used to specify that data is exported either with or without its security attributes.

b) Translation of exported data (FCO_TED) When a subject exports data and/or security attributes, this data and/or security attributes may need translation to allow consistent interpretation. An example would be translating the username "root" from a non-Unix system to a

different username for a Unix system, as “root” has a special meaning in a Unix system.

c) When a subject exports data to a user, the TSF may provide:

1) Availability to the user, which may be specified through the Availability of exported data (FCO_AED) family;

2) Confidentiality between subject and user, which may be specified through the Confidentiality of exported data (FCO_CED) family;

3) Integrity between subject and user, which may be specified through the Integrity of exported data (FCO_IED) family.

d) Non-repudiation of exported data (FCO_NRE) When a subject exports data to a user:

1) the TSF may add evidence that may be used to prove that data indeed comes from that subject;

2) the TSF may request evidence from the user that may later be used to prove that the user received the data.

e) Unobservability of export (FCO_UNE) When a subject exports data to a user:

1) the TSF may hide to which user the data is being exported;

2) the TSF may hide that it is exporting data.

11.2 Import families

a) Import from outside TSF control (FCO_ITC) This family may be used to limit a subject from importing all data that the user is sending to the subject. An example would be a TOE that is used to store logging data from other IT. As the TOE fills up, Import from outside TSF control (FCO_ITC) may be used to specify that the TOE starts refusing less important logging data. In addition, this family may be used to specify that data is imported either with or without its security attributes.

b) Translation of imported data (FCO_TID) When a subject imports data and/or security attributes, this data and/or security attributes may need translation to allow consistent interpretation. An example would be interpreting and summarising an X509 certificate into a form meaningful to the TOE.

c) When a subject imports data from a user, the TSF may provide/assist with:

1) Confidentiality between user and subject, which may be specified through the Confidentiality of imported data (FCO_CID)family;

2) Integrity between user and subject, which may be specified through the Integrity of imported data (FCO_IID) family.

d) Non-repudiation of imported data (FCO_NRI) When a subject imports data from a user:

1) the TSF may request evidence from the user that the TSF may later use to prove that the data indeed came from that user;

2) the TSF may generate evidence that the subject indeed received that data and send this evidence to the user.

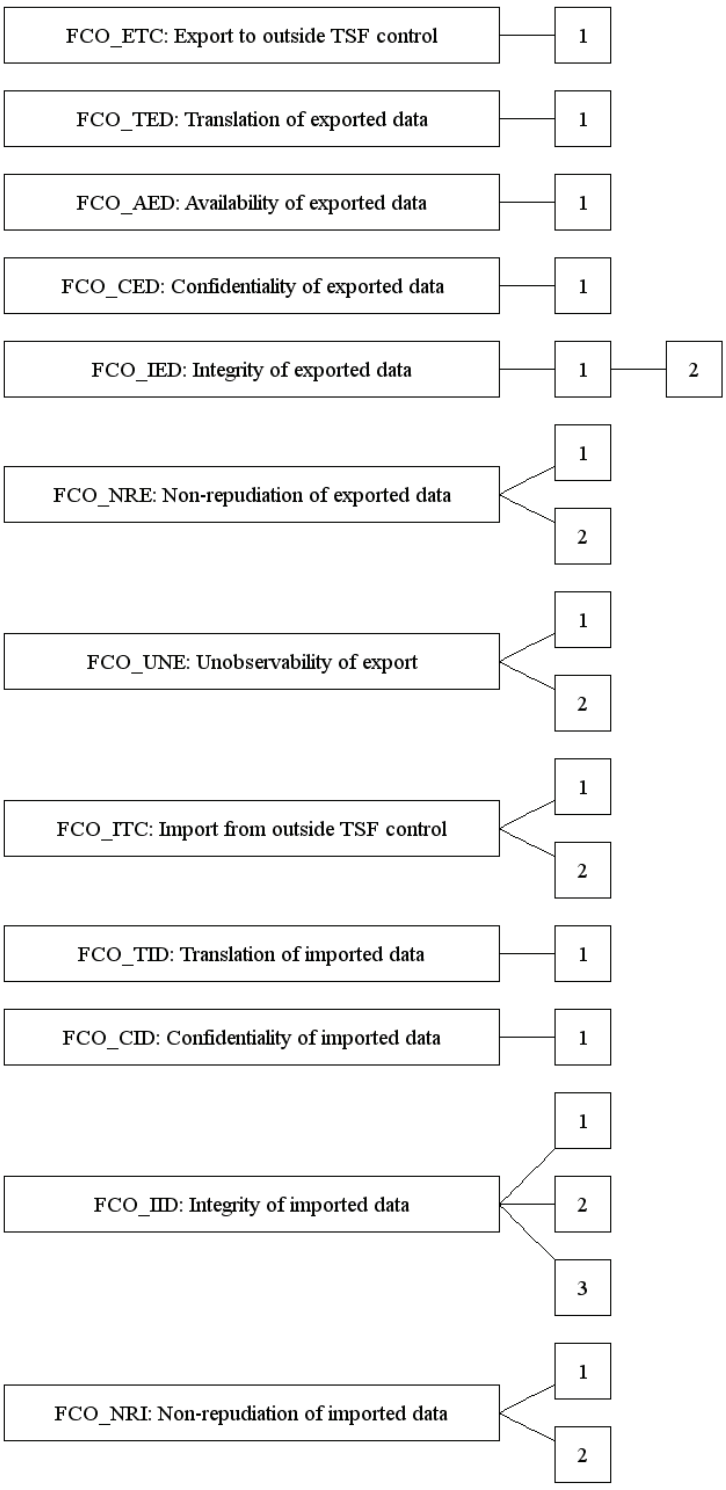


Figure 15 - FCO: Communication class decomposition

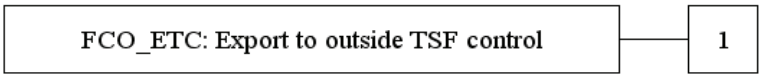
11.3 Export to outside TSF control (FCO_ETC)

227 This family may be used to limit a subject from exporting all data to a user that the subject has access to. In addition, this family may be used to specify that data is exported either with or without (some of) its associated security attributes.

228 The PP/ST writer may consider using components from the Translation of exported data (FCO_TED) family to ensure that the user which the data and security attributes are exported to, may understand this data and security attributes.

229 Especially when security attributes are exported, the PP/ST writer may also consider using components from the Integrity of exported data (FCO_IED) family to protect the integrity of these security attributes once they leave the TSF.

Component levelling



Associated operations: FCO_ETC.1

230 None.

Audit: FCO_ETC.1

231 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful export, possibly with the data and/or security attributes that were (were not) exported.

FCO_ETC.1 Export of data and/or security attributes

Hierarchical to: No other components.

Dependencies: No dependencies.

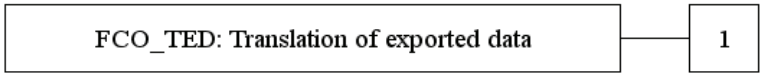
FCO_ETC.1.1 **The TSF shall enforce [assignment: *rules on whether export is allowed*] when [assignment: *subject*] exports [assignment: *list of data and/or security attributes*] to a user bound to that subject.**

11.4 Translation of exported data (FCO_TED)

232 When a subject exports data and/or security attributes, this data and/or security attributes may need translation to allow consistent interpretation. The Translation of exported data (FCO_TED) family should be used to specify these rules.

233 Note that Translation of exported data (FCO_TED) is only intended for specification of translation rules that have relevancy for the security of the user that is being exported to. Translation of exported data (FCO_TED) is not intended to specify all data translations or entire protocols.

Component levelling



Associated operations: FCO_TED.1

234 None.

Audit: FCO_TED.1

235 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each translation, possibly with the values of the data and security attributes that were translated.

FCO_TED.1 Translation of exported data

Hierarchical to: No other components.

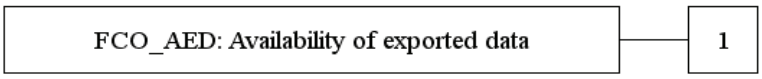
Dependencies: No dependencies.

FCO_TED.1.1 **The TSF shall apply [assignment: *rules on how data and security attributes are translated*] when [assignment: *subject*] exports [assignment: *list of data and/or security attributes*] to a user bound to that subject.**

11.5 Availability of exported data (FCO_AED)

236 This family defines the rules for the prevention of loss of availability of data being exported from a subject to a user.

Component levelling



Associated operations: FCO_AED.1

237 None.

Audit: FCO_AED.1

238 There are no auditable events foreseen.

FCO_AED.1 Availability of exported data

Hierarchical to: No other components.

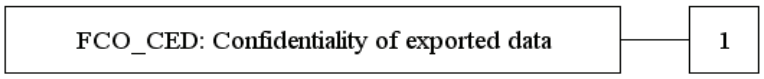
Dependencies: No dependencies.

FCO_AED.1.1 **The TSF shall ensure the availability of [assignment: *list of data and/or security attributes*] provided by [assignment: *subject*] to a user bound to that subject within [assignment: *a defined availability metric*] given the following conditions [assignment: *conditions to ensure availability*].**

11.6 Confidentiality of exported data (FCO_CED)

239 This family defines the rules for the protection of confidentiality of data being exported from a subject to a user.

Component levelling



Associated operations: FCO_CED.1

240 None.

Audit: FCO_CED.1

241 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each export of data, possibly accompanied by references to the data that was transferred.

FCO_CED.1 Confidentiality of exported data

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_CED.1.1 **The TSF shall protect the confidentiality of [assignment: *list of data and/or security attributes*] provided by [assignment: *subject*] to a user bound to that subject.**

11.7 Integrity of exported data (FCO_IED)

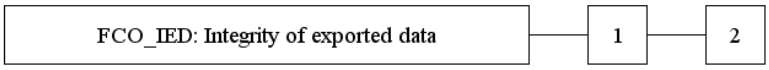
242 This family defines the rules for the protection of integrity of data being exported from a subject to a user.

243 The family consists of two components:

a) FCO_IED.1 Integrity of exported data without recovery, where the TSF provides the means for the user to detect integrity anomalies (e.g. by adding a hash) in data that is being exported to it.

b) FCO_IED.2 Integrity of exported data with assisted recovery, where the TSF not only provides a means for detection of integrity anomalies, but, once an integrity anomaly is detected, assists in recovery. This could be done by re-sending the data, or adding error-correcting codes to the data.

Component levelling



Associated operations: FCO_IED.1, FCO_IED.2

244 None.

Audit: FCO_IED.1

245 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each export of data, possibly accompanied by references to the data that was transferred

Audit: FCO_IED.2

246 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each export of data, possibly accompanied by references to the data that was transferred

b) Each successful or unsuccessful recovery of an integrity anomaly

FCO_IED.1 Integrity of exported data without recovery

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_IED.1.1 **When [assignment: *subject*] transmits [assignment: *list of data and/or security attributes*] to a user bound to that subject, the TSF shall provide**

that user the means to detect [selection: *modification, deletion, insertion, replay, [assignment: other integrity]*] anomalies.

FCO_IED.2 Integrity of exported data with assisted recovery

Hierarchical to: FCO_IED.1 Integrity of exported data without recovery

Dependencies: No dependencies.

FCO_IED.2.1 When [assignment: *subject*] transmits [assignment: *list of data and/or security attributes*] to a user bound to that subject, the TSF shall provide that user the means to detect [selection: *modification, deletion, insertion, replay, [assignment: other integrity]*] anomalies.

FCO_IED.2.2 **The TSF shall be able to assist the user in recovering the data and/or security attributes.**

11.8 Non-repudiation of exported data (FCO_NRE)

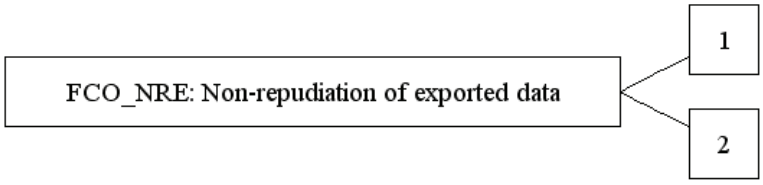
247 This family provides requirements for evidence that data has been exported. This evidence may later be used to prove that the data was indeed exported.

248 This family consists of two components:

a) FCO_NRE.1 Non-repudiation of receipt of exported data, where the TSF requires evidence from the user that the user received exported data. This evidence may later be used to prove that that user indeed received the exported data;

b) FCO_NRE.2 Non-repudiation of origin of exported data, where the TSF provides evidence to a user that the data that the user has received has been exported by a subject. This evidence may later be used by the user to prove that that subject indeed exported that data.

Component levelling



Associated operations: FCO_NRE.1, FCO_NRE.2

249 None.

Audit: FCO_NRE.1

250 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The subject and user exporting the data;

b) The data (or some characterisation of it).

c) The receipt of evidence;

d) The actions undertaken.

Audit: FCO_NRE.2

251 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The subject and user exporting the data;

b) The data (or some characterisation of it);

FCO_NRE.1 Non-repudiation of receipt of exported data

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control
FCO_IED.1 Integrity of exported data without recovery

FCO_NRE.1.1 **The TSF shall require evidence that [assignment: *subject*] has exported [assignment: *data*] to a user bound to that subject and that that user has received the data.**

FCO_NRE.1.2 **The TSF will store this evidence in [assignment: *object*].**

FCO_NRE.1.3 **If no evidence is received within [assignment: *time*], the TSF shall [assignment: *list of actions*].**

FCO_NRE.2 Non-repudiation of origin of exported data

Hierarchical to: No other components.

Dependencies: FCO_IED.1 Integrity of exported data without recovery

FCO_NRE.2.1 **The TSF shall generate evidence that [assignment: *subject*] has exported [assignment: *data*] to a user bound to that subject.**

FCO_NRE.2.2 **The TSF shall export this evidence to that user to which the data was exported.**

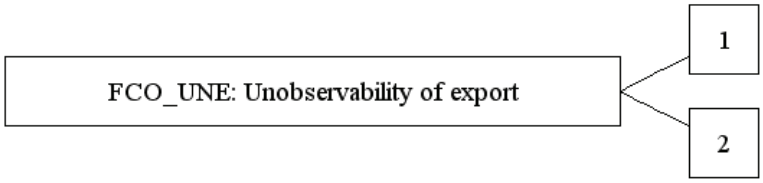
11.9 Unobservability of export (FCO_UNE)

252 This family defines the rules for hiding the act of exporting data to a user. This is different from hiding the data that has been exported, which should be specified by Confidentiality of exported data (FCO_CED).

253 This hiding of export may be done by hiding the identity of the user that the data is being exported to. A possible implementation is broadcasting it over Ethernet or radio, so that other users are unable to determine who is actually receiving and using the data. This should be specified with FCO_UNE.1 Unobservability of the user of exported data.

254 This hiding of export may also be done by hiding the fact that export is taking place at all. A possible implementation is padding of an encrypted data stream when nothing is being exported. As the TOE is constantly exporting something, other users are unable to determine that actual data is being exported. This should be specified with FCO_UNE.2 Unobservability of export.

Component levelling



Associated operations: FCO_UNE.1, FCO_UNE.2

255 None.

Audit: FCO_UNE.1, FCO_UNE.2

256 There are no auditable events foreseen.

FCO_UNE.1 Unobservability of the user of exported data

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_UNE.1.1 **The TSF shall export [assignment: *list of data and/or security attributes*] to a user bound to [assignment: *subject*] in such a way that it cannot be observed to which user export is taking place.**

FCO_UNE.2 Unobservability of export

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_UNE.2.1 **The TSF shall export [assignment: *list of data and/or security attributes*] to a user bound to [assignment: *subject*] in such a way that it cannot be observed whether that data and/or security attributes are being exported.**

11.10 Import from outside TSF control (FCO_ITC)

257 This family may be used to limit a subject from importing data that a bound user is sending to the subject. An example would be a TOE that is used to store logging data from other IT. As the TOE fills up, Import from outside TSF control (FCO_ITC) may be used to specify that the TOE starts refusing less important logging data.

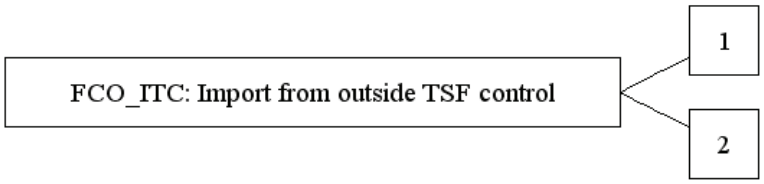
258 In addition, this family may be used to specify that data is imported either with or without its security attributes.

259 Importing data and/or security attributes transport these data and/or security attributes to the subject importing them. This subject may then act on this, through application of the Access control (FDP_ACC) family, e.g. by creating a new object with that data and security attributes.

260 The PP/ST writer may consider using components from the Translation of imported data (FCO_TID) family to ensure that the subject importing the data and security attributes are exported to may understand this data and security attributes.

261 Especially when security attributes are imported, the PP/ST writer may also consider using components from the Integrity of imported data (FCO_IID) family to protect the integrity of these security attributes between the user and the subject.

Component levelling



Associated operations: FCO_ITC.1, FCO_ITC.2

262 None.

Audit: FCO_ITC.1

263 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful import, possibly with the data that was (not) imported.

Audit: FCO_ITC.2

264 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful import, possibly with the data that was (not) imported;

b) Successful/unsuccessful import, possibly with the data that was (not) imported and/or values of the security attributes that were (not) imported.

FCO_ITC.1 Import without security attributes

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_ITC.1.1 **The TSF shall enforce [assignment: *rules on whether import is allowed*] when [assignment: *subject*] imports [assignment: *list of data*] from a user bound to that subject.**

FCO_ITC.1.2 **The data shall be imported without security attributes.**

FCO_ITC.2 Import with security attributes

Hierarchical to: No other components.

Dependencies: [FCO_IID.1 Integrity of imported data without recovery, or
FCO_IID.2 Integrity of imported data with assisted recovery]

FCO_ITC.2.1 **The TSF shall enforce [assignment: *rules on whether import is allowed*] when [assignment: *subject*] [assignment: *list of data*] from a user bound to that subject.**

FCO_ITC.2.2 **The imported data shall be imported with the security attributes [assignment: *security attributes*].**

FCO_ITC.2.3 **The TSF shall associate the security attributes with the imported data.**

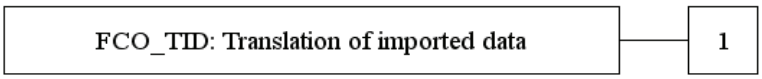
11.11 Translation of imported data (FCO_TID)

265 When a subject imports data and/or security attributes, this data and/or security attributes may need translation to allow consistent interpretation. The Translation of imported data (FCO_TID) family should be used to specify these rules.

266 An example would be data belonging to a user "root" on a non-Unix system being imported into a Unix-based TOE. As "root" has a specific meaning in Unix, this username may have to be translated to another term.

267 Note that Translation of imported data (FCO_TID) is only intended for specification of translation rules that have relevancy for the TSP. Translation of imported data (FCO_TID) is not intended to specify all translations or entire protocols.

Component levelling



Associated operations: FCO_TID.1

268 None.

Audit: FCO_TID.1

269 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each translation, possibly with the values of the data and security attributes that were translated.

FCO_TID.1 Translation of imported data

Hierarchical to: No other components.

Dependencies: No dependencies.

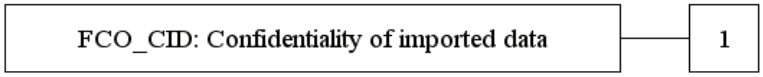
FCO_TID.1.1 **The TSF shall apply [assignment: *rules on how data and security attributes are translated*] when [assignment: *subject*] imports [assignment: *list of data and/or security attributes*] from a user bound to that subject.**

11.12 Confidentiality of imported data (FCO_CID)

270 This family defines the rules for the protection of confidentiality of data being imported by a subject from a user.

271 Note that the TOE may only assist the user in protecting confidentiality (e.g. by offering decryption or a physically secured interface), but it is unable to guarantee confidentiality by itself. In other words, once the data is imported the TSF may guarantee confidentiality, but outside the TOE, the responsibility for confidentiality lies with the user and the TSF may only assist the user in providing this confidentiality.

Component levelling



Associated operations: FCO_CID.1

272 None.

Audit: FCO_CID.1

273 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each import of data, possibly accompanied by reference to the data that was imported.

FCO_CID.1 Confidentiality of imported data

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_CID.1.1 **The TSF shall assist in protecting the confidentiality of [assignment: *list of data and/or security attributes*] provided to [assignment: *subject*] by a user bound to that subject.**

11.13 Integrity of imported data (FCO_IID)

274 This family defines the rules for the protection of integrity of data being imported by a subject from a user.

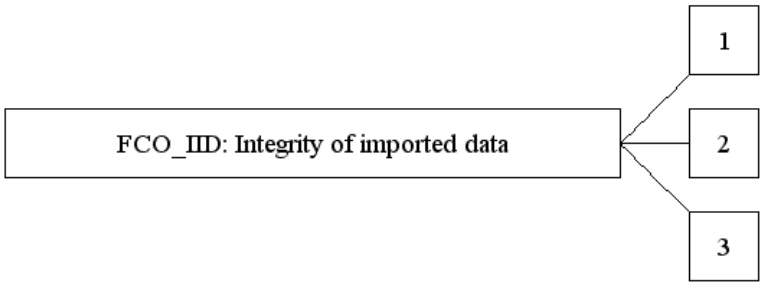
275 This family has of three components:

a) FCO_IID.1 Integrity of imported data without recovery, where the TOE need only detect that the imported data has integrity anomalies;

b) FCO_IID.2 Integrity of imported data with assisted recovery, where the TOE may detect and, with assistance of the user, repair anomalies, e.g. by asking the user to resubmit the data;

c) FCO_IID.3 Integrity of imported data with unassisted recovery, where the TOE may detect and repair anomalies without assistance of the user, e.g. by using redundancy in the data.

Component levelling



Associated operations: FCO_IID.1, FCO_IID.2, FCO_IID.3

276 None.

Audit: FCO_IID.1

277 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each detected anomaly;

b) Each import of data, possibly accompanied by reference to the data that was imported.

Audit: FCO_IID.2, FCO_IID.3

278 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Each detected anomaly and whether it was repaired or not;

b) Each import of data, possibly accompanied by reference to the data that was imported.

FCO_IID.1 Integrity of imported data without recovery

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_IID.1.1 **The TSF shall monitor the integrity of [assignment: *list of data and/or security attributes*] provided to [assignment: *subject*] by a user bound to that subject for [selection: *modification, deletion, insertion, replay*] anomalies.**

FCO_IID.1.2 **On detection of an anomaly the TSF shall discard the data and/or security attributes.**

FCO_IID.2 Integrity of imported data with assisted recovery

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_IID.2.1 **The TSF shall monitor the integrity of [assignment: *list of data and/or security attributes*] provided to [assignment: *subject*] by a user bound to that subject against [selection: *modification, deletion, insertion, replay*] anomalies.**

FCO_IID.2.2 **The TSF shall be able to recover the data and/or security attributes with assistance of the user when an integrity anomaly has occurred.**

FCO_IID.3 Integrity of imported data with unassisted recovery

Hierarchical to: No other components.

Dependencies: No dependencies.

FCO_IID.3.1 **The TSF shall monitor the integrity of [assignment: *list of data and/or security attributes*] provided to [assignment: *subject*] by a user bound to that subject against [selection: *modification, deletion, insertion, replay*] anomalies.**

FCO_IID.3.2 **The TSF shall be able to recover the data and/or security attributes without assistance of the user when an integrity anomaly has occurred.**

11.14 Non-repudiation of imported data (FCO_NRI)

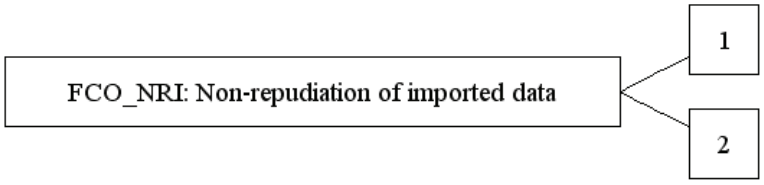
279 This family provides requirements for evidence that data has been imported. This evidence may later be used to prove that the data was indeed imported.

280 This family consists of two components:

a) FCO_NRI.1 Non-repudiation of receipt of imported data, where the TSF provides evidence to a user that a subject imported data from that user. This evidence may later be used by the user to prove that the user indeed imported the data to the subject.

b) FCO_NRI.2 Non repudiation of origin of imported data, where the TSF requires evidence from a user that data imported by a subject from that user is indeed is from that user. This evidence may later be used to prove that the user indeed imported the data to the subject.

Component levelling



Associated operations: FCO_NRI.1, FCO_NRI.2

281 None.

Audit: FCO_NRI.1

282 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The subject and user importing the data;

b) The data (or some characterisation of it).

Audit: FCO_NRI.2

283 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The subject and user importing the data;

b) The data (or some characterisation of it);

c) The receipt of evidence;

d) The actions undertaken.

FCO_NRI.1 Non-repudiation of receipt of imported data

Hierarchical to: No other components.

Dependencies: FCO_IED.1 Integrity of exported data without recovery
[FCO_IID.1 Integrity of imported data without recovery, or
FCO_IID.2 Integrity of imported data with assisted recovery, or
FCO_IID.3 Integrity of imported data with unassisted recovery]

FCO_NRI.1.1 **The TSF shall generate evidence that [assignment: *subject*] has imported [assignment: *data*] from a user bound to that subject.**

FCO_NRI.1.2 **The TSF shall export this evidence to the user to which the data was imported.**

FCO_NRI.2 Non repudiation of origin of imported data

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control
[FCO_IID.1 Integrity of imported data without recovery, or
FCO_IID.2 Integrity of imported data with assisted recovery, or
FCO_IID.3 Integrity of imported data with unassisted recovery]

FCO_NRI.2.1 **The TSF shall require evidence that [assignment: *subject*] has imported [assignment: *data*] from a user bound to that subject and that that user has sent the data.**

FCO_NRI.2.2 **The TSF will store this evidence in [assignment: *object*].**

FCO_NRI.2.3 **If no evidence is received within [assignment: *time*] the TSF shall [assignment: *list of actions*].**

12 Class FAU: Security audit

284 Security auditing involves recognising, recording, storing, and analysing information related to security relevant activities (i.e. activities controlled by the TSP). The resulting audit records may be examined to determine which security relevant activities took place and which subject or user is responsible for them.

285 Note that the FAU class does not cover protection of the audit log. As this audit log is an object, access to it should be specified using Access control (FDP_ACC) and overflow should be specified using Resource allocation (FPT_RSA).

286 This class consists of the following three families:

a) Audit data is generated and stored in an object according to rules (Security audit data generation (FAU_GEN));

b) The TSF may monitor the audited events for potential security violations (Security audit analysis (FAU_SAA));

c) When a potential security violation is noticed, the TOE may act on it (Security audit automatic response (FAU_ARP)).

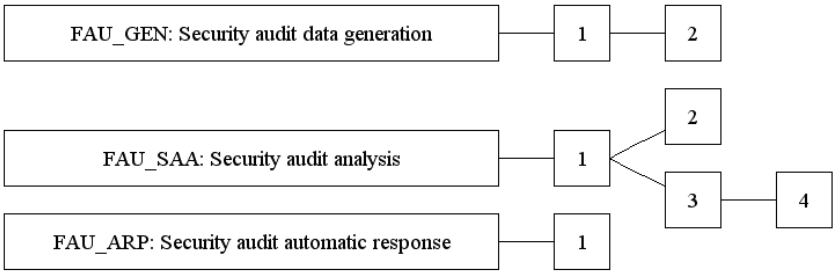


Figure 16 - FAU: Security audit class decomposition

12.1 Security audit data generation (FAU_GEN)

287 This family defines requirements for recording the occurrence of security relevant events that take place under TSF control. This family can be used to enumerate the types of events that are recorded by the TSF, and to identify the audit-related information that should be provided within various types of audit record.

288 If all subjects may arbitrarily access the object in which the audited events are stored, audit may provide little value. The PP/ST author should therefore use Access control (FDP_ACC) to regulate access to that object. Similarly, the filling up and overflow of that object may cause problems, so the PP/ST author should use FPT_RSA.1 Maximum quotas for subjects and objects to describe this case.

289 In some TOEs the audit data is not stored in the TOE, but exported to and stored by a user (usually a machine user). In this case, the PP/ST writer should define a subject, use Access control (FDP_ACC) to specify that that subject may read and remove data from the object where the audit data is temporarily stored, and use Export to outside TSF control (FCO_ETC) to export the data to the user.

290 Special care should be taken when “shutdown of the audit functions” is included as an auditable event. If the TSF can somehow be suddenly terminated or interrupted (e.g. power loss) it may prove impossible or very impractical to meet this requirement.

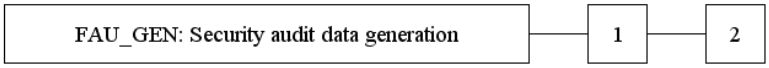
291 If it is important to be able to link the identity of users to certain events, this identity should be represented as a security attribute of the subject(s) causing that event. This can be specified though the User registration (FIA_URE) and User-subject binding (FIA_USB) families.

292 This family consists of two components:

a) FAU_GEN.1 Audit data generation without time, which allows the generation of audit data without recording the time that the events occur. FAU_GEN.1 Audit data generation without time is useful for a TOE that operates in an Operational Environment that has no access to a trusted time source (e.g. a smart card in a smart card reader) or when it is not necessary to know the exact time of an event.

b) FAU_GEN.2 Audit data generation with time, which ensures that all events are logged with the time that they occur. To ensure that the TOE is using the correct time, there is a dependency on either FMI_TIM.1 Time stamps (when the TOE generates its own time) or FCO_ITC.1 Import without security attributes and FCO_IID.1 Integrity of imported data without recovery (for when the TOE imports time).

Component levelling



Associated operations: FAU_GEN.1, FAU_GEN.2

293 None.

Audit: FAU_GEN.1, FAU_GEN.2

294 There are no auditable events foreseen.

FAU_GEN.1 Audit data generation without time

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control
FPT_RSA.1 Maximum quotas for subjects and objects

FAU_GEN.1.1 **The TSF shall store an audit record in [assignment: *object*] of the following events: [selection: *start-up of the audit functions, shut-down of the audit functions, [assignment: rules for which other events will be audited]*].**

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

a) Type of event, values of [assignment: *security attributes*] of the subject(s) associated with the event, the [selection: *success, failure, [assignment: other outcome(s)]*] of the event; and

b) [assignment: *other information*].

FAU_GEN.2 Audit data generation with time

Hierarchical to: FAU_GEN.1 Audit data generation without time

Dependencies: [FMI_TIM.1 Time stamps, or
(FCO_ITC.1 Import without security attributes, and
FCO_IID.1 Integrity of imported data without recovery)]
FDP_ACC.1 Access control
FPT_RSA.1 Maximum quotas for subjects and objects

FAU_GEN.2.1 The TSF shall store an audit record in [assignment: *object*] of the following events: [selection: *start-up of the audit functions, shut-down of the audit functions, [assignment: rules for which other events will be audited]*].

FAU_GEN.2.2 The TSF shall record within each audit record the following information:

a) **Date and time** of **the** event, **type of event,** values of [assignment: *security **attributes** of the **subject***], the [selection: *success, failure, [assignment: other outcome(s)]*] of the event; and

b) [assignment: *other information*].

12.2 Security audit analysis (FAU_SAA)

295 This family defines requirements for automated means that analyse audit records looking for possible, actual, or impending violations of the TSP. This analysis may work in support of intrusion detection, or automatic response to an imminent TSP violation.

296 The actions to be taken based on the detection should be specified using the Security audit automatic response (FAU_ARP) family.

297 This family consists of four components:

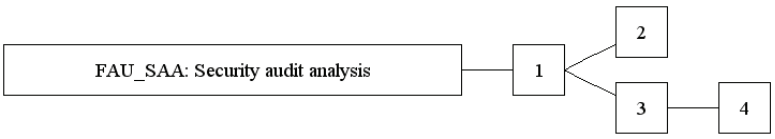
– In FAU_SAA.1 Potential violation analysis, basic threshold detection on the basis of a fixed rule set is required.

– In FAU_SAA.2 Profile based anomaly detection, the TSF maintains individual profiles of system usage, where a profile represents the historical patterns of usage performed by members of the profile target group. A profile target group refers to a group of one or more individuals (e.g. a single user, users who share a group ID or group account, users who operate under an assigned role, users of an entire system or network node) who interact with the TSF. Each member of a profile target group is assigned an individual suspicion rating that represents how well that member's current activity corresponds to the established patterns of usage represented in the profile. This analysis may be performed at runtime or during a post-collection batch-mode analysis.

– In FAU_SAA.3 Simple attack heuristics, the TSF shall be able to detect the occurrence of signature events that represent a significant threat to TSP enforcement. This search for signature events may occur in real-time or during a post-collection batch-mode analysis.

– In FAU_SAA.4 Complex attack heuristics, the TSF shall be able to represent and detect multi-step intrusion scenarios. The TSF is able to compare system events (possibly performed by multiple individuals) against event sequences known to represent entire intrusion scenarios. The TSF shall be able to indicate when a signature event or event sequence is found that indicates a potential violation of the TSP.

Component levelling



Associated operations: FAU_SAA.1, FAU_SAA.2, FAU_SAA.3, FAU_SAA.4

298 None.

Audit: FAU_SAA.1, FAU_SAA.2, FAU_SAA.3, FAU_SAA.4

299 There are no auditable events foreseen.

FAU_SAA.1 Potential violation analysis

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation without time

FAU_SAA.1.1 **The TSF shall apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the TSP.**

FAU_SAA.1.2 **The TSF shall enforce the following rules for monitoring audited events:**

a) Accumulation or combination of [assignment: *subset of defined auditable events*] known to indicate a potential TSP violation;

b) [assignment: *any other rules*].

FAU_SAA.2 Profile based anomaly detection

Hierarchical to: FAU_SAA.1 Potential violation analysis

Dependencies: FAU_GEN.1 Audit data generation without time

FAU_SAA.2.1 The TSF shall **maintain profiles** of **usage, where an individual profile represents** the **historical patterns** of **usage performed by** the **member(s)of [assignment: *the profile target group*].**

FAU_SAA.2.2 The TSF shall **maintain a suspicion rating associated with each subject whose activity is recorded in a profile, where** the **suspicion rating represents the degree to which the subject's current activity is found inconsistent with the established patterns** of **usage represented in the profile.**

FAU_SAA.2.3 **The TSF shall indicate an imminent violation of the TSP when a subject's suspicion rating exceeds the following threshold conditions [assignment: *conditions under which anomalous activity is reported by the TSF*].**

FAU_SAA.3 Simple attack heuristics

Hierarchical to: FAU_SAA.1 Potential violation analysis

Dependencies: FAU_GEN.1 Audit data generation without time

FAU_SAA.3.1 The TSF shall **maintain an internal representation** of the **following signature** events **[assignment: *a subset of system events*] that may** indicate a violation of the TSP.

FAU_SAA.3.2 The TSF shall **compare** the **signature** events **against the record** of **system activity discernible from an examination of** [assignment: ***the information to be used to determine system activity***]**.**

FAU_SAA.3.3 **The TSF shall indicate an imminent violation of the TSP when a system event is found to match a signature event that indicates a potential violation of the TSP.**

FAU_SAA.4 Complex attack heuristics

Hierarchical to: FAU_SAA.3 Simple attack heuristics

Dependencies: No dependencies.

FAU_SAA.4.1 The TSF shall maintain an internal representation of the following **event sequences of known intrusion scenarios [assignment: *list of sequences of system events whose occurrence are representative of known penetration scenarios*] and the following** signature events [assignment: *a subset of system events*] that may indicate a **potential** violation of the TSP.

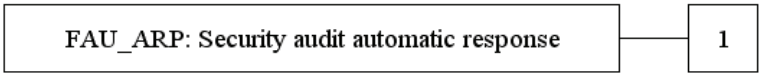
FAU_SAA.4.2 The TSF shall compare the signature events **and event sequences** against the record of system activity discernible from an examination of [assignment: *the information to be used to determine system activity*].

FAU_SAA.4.3 The TSF shall indicate an imminent violation of the TSP when system **activity** is found to match a signature event **or event sequence** that indicates a potential violation of the TSP.

12.3 Security audit automatic response (FAU_ARP)

300 This family defines the response to be taken in case of a potential violation of the TSP as detected through components of the Security audit analysis (FAU_SAA) family.

Component levelling



Associated operations: FAU_ARP.1

301 None.

Audit: FAU_ARP.1

302 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Actions undertaken.

FAU_ARP.1 Security audit automatic response

Hierarchical to: No other components.

Dependencies: FAU_SAA.1 Potential violation analysis

FAU_ARP.1.1 The TSF shall [assignment: *list of actions*] upon detection of a potential violation of the TSP.

13 Class FPT: Protection of the TSF

303 Other functional classes describe protection provided by the TSF, but assume that the TSF itself is physically protected, that TSF integrity is always maintained and that the TSF has an infinite amount of idealised resources at its disposal.

304 This class defines functional families of functional requirements that address the protection of the TSF and its resources. The class consists of three groups of families:

a) Testing, failure and recovery, where the TSF

- may test itself (TSF self test (FPT_TST));
- may test its machine users (Testing of users (FPT_TOU));
- will continue to operate when certain failures occur (Fault tolerance (FPT_FLT));
- will continue to operate partially (or not all) when certain failures occur (Fail secure (FPT_FLS));
- may recover from certain failures (Trusted recovery (FPT_RCV)).

b) Physical protection, where the TSF may be tamper-responsive or tamper-resistant or allow detection of tampering (TSF physical protection (FPT_PHP)).

c) Resource use, where the TSF

- will prioritise its operations when resources needed by the TSF are overtaxed so that the scarce resources will be allocated to high-priority operations and shall not be monopolised by lower priority operations (Priority (FPT_PRI));
- may set limits to the amount of resources consumed by objects and subjects (Resource allocation (FPT_RSA));
- may permanently remove data from resources, so that when these resources are ever access or re-used later on, the data shall not be retrievable (Residual information protection (FPT_RIP)).

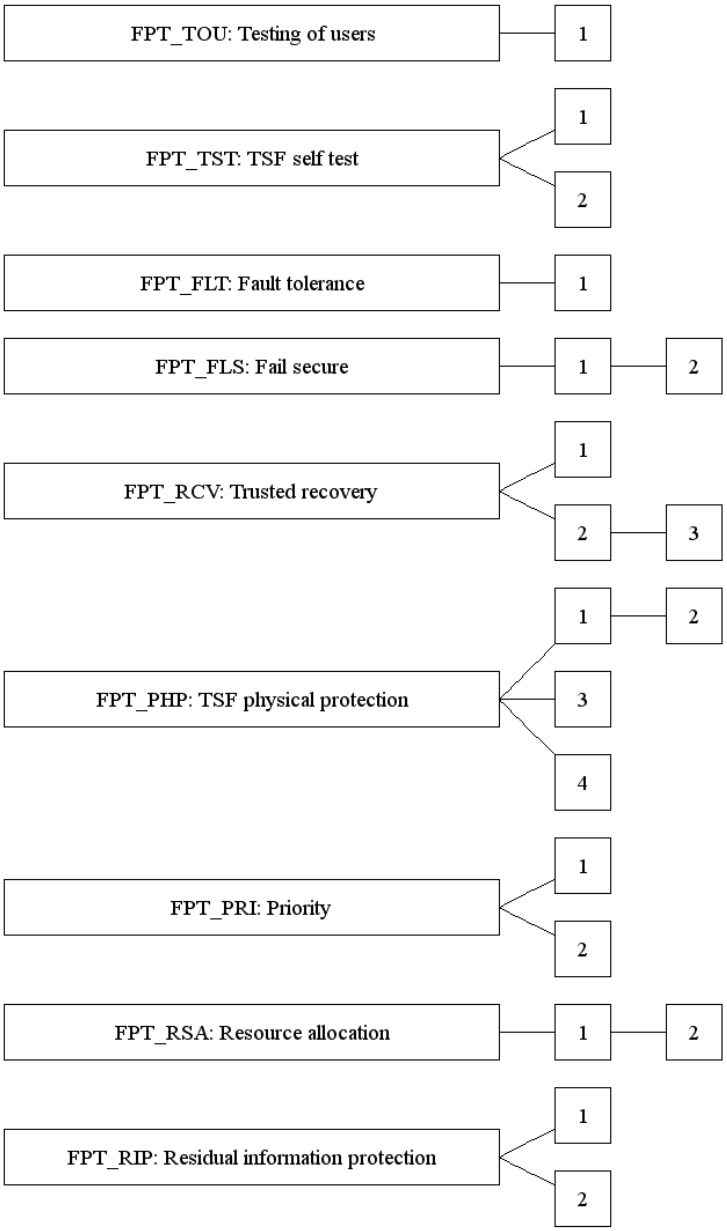


Figure 17 - FPT: Protection of the TSF class decomposition

13.1 Testing of users (FPT_TOU)

305 This family defines requirements for the TSF to perform testing one or more of its users.

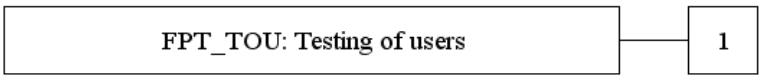
306 The users in this requirement may be machine users or human users but this requirement is intended to apply to machine users. It may make no sense when applied to human users.

307 Note that users are active entities outside the TOE. They may be applications running on the TOE, hardware or software running "underneath" the TOE (platforms, operating systems etc.) or applications/boxes connected to the TOE (intrusion detection systems, firewalls, login servers, time servers etc.

308 Examples of the types of tests that may be run are:

- Tests for the presence of a firewall, and possibly whether it is correctly configured;
- Tests of some of the security properties of the operating system that an application TOE runs on;
- Test of some of the security properties of the IC that a smartcard OS TOE runs on.

Component levelling



Associated operations: FPT_TOU.1

309 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Run_Operational_Environment_Tests: an operation that runs the suite of tests defined in **FPT_TOU.1.1**. This operation should be applied to the TSF.

Audit: FPT_TOU.1

310 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Execution of the tests, with overall or detailed results.

FPT_TOU.1 Operational environment testing

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_TOU.1.1 **The TSF shall run a suite of tests [selection: *during each start-up, periodically during normal operation, at the request of a subject, [assignment: other conditions]*] to determine the correct operation of [assignment: *user*].**

13.2 TSF self test (FPT_TST)

311 The family defines requirements for the TSF to test itself, and verify the integrity of the TSF's stored executable code and/or objects. The actions to be taken by the TOE as the result of this testing are defined in the Fail secure (FPT_FLS) family.

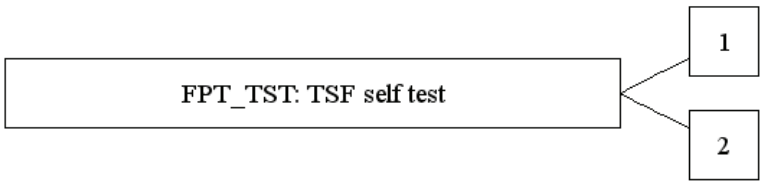
312 The requirements of this family are needed to detect the corruption of the TSF or objects by various failures that do not necessarily stop the TSF operation (which would be handled by other families). Such failures may occur either because of unforeseen failure modes or associated oversights in the design of hardware, firmware, or software, or because of malicious corruption of the TSF.

313 The family consists of two components:

a) FPT_TST.1 TSF self-testing which tests the TSF itself

b) FPT_TST.2 Integrity testing which verifies the integrity of TSF stored executable code and objects.

Component levelling



Associated operations: FPT_TST.1

314 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Run_Self_Tests: an operation that runs the suite of tests defined in **FPT_TST.1.1**. This operation should be applied to the TSF.

Associated operations: FPT_TST.2

315 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Run_Integrity_Tests: an operation that runs the suite of tests defined in **FPT_TST.2.1**. This operation should be applied to the TSF.

Audit: FPT_TST.1

316 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Execution of the tests, with overall or detailed results.

Audit: FPT_TST.2

317 There are no auditable events foreseen.

FPT_TST.1 TSF self-testing

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_TST.1.1 **The TSF shall run a suite of tests [selection: *immediately after installation, during each start-up, periodically during normal operation, at the request of a subject, [assignment: other conditions]*] to demonstrate the correct operation of [selection: *[assignment: parts of the TSF] , the TSF*].**

FPT_TST.2 Integrity testing

Hierarchical to: No other components.

Dependencies: No dependencies.

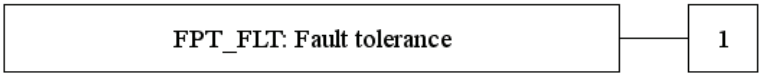
FPT_TST.2.1 **The TSF shall run a suite of tests [selection: *immediately after installation, during each start-up, periodically during normal operation, at the request of a subject, [assignment: other conditions]*] to verify the integrity of [selection: *the stored executable code of the TSF, [assignment: list of objects]*].**

13.3 Fault tolerance (FPT_FLT)

318 The requirements of this family ensure that the TSF will maintain correct operation even in the event of failures. Using Fault tolerance (FPT_FLT) means that the TSF will continue to meet all other SFRs in the ST in case of failure.

319 The case that the TSF no longer meets all SFRs or that the TSF meets different SFRs after failure should be specified through the Fail secure (FPT_FLS) family.

Component levelling



Associated operations: FPT_FLT.1

320 None.

Audit: FPT_FLT.1

321 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Failures of the TSF, possibly with the type of the failure.

FPT_FLT.1 Fault tolerance

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_FLT.1.1 **The TSF shall continue to meet the TSP when the following failures occur: [assignment: *list of failures or types of failures*].**

13.4 Fail secure (FPT_FLS)

322 This family provides requirements for the TSF to enter failure mode when failures occur. The entering of failure mode may occur:

a) Implicitly: certain failures occur and as a result of these failures, failure mode is entered;

b) Explicitly: the TSF tests itself (see FPT_TST.1 TSF self-testing), determines that a failure has occurred and as a result decides to enter failure mode.

323 This failure mode is specified by listing:

a) All SFRs that no longer apply in failure mode;

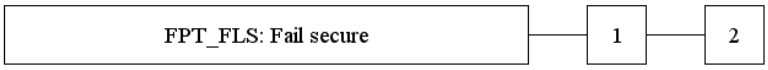
b) Additional SFRs that did not apply in non-failure mode, but apply in failure mode.

324 Conceptually, when Fail secure (FPT_FLS) is included in a PP/ST, the TSP is split into two sub-TSPs: a "non-failure mode" TSP that normally applies and a "failure mode" TSP that applies only when the TSF fails.

325 The “failure mode” TSP may include any SFR, but SFRs for terminating existing bindings of users (FIA_TOB.1 TSF-initiated termination of binding), restricting new bindings (TSF binding rules (FIA_TBR)) and restricting the abilities of bound subjects (Access control (FDP_ACC)) may be more applicable than others.

326 Once the TSF enters failure mode, this may be permanent, or it may be possible to recover to non-failure mode. This should be specified by including an SFR based on the Trusted recovery (FPT_RCV) family in the set of SFRs that applies during failure mode.

Component levelling



Associated operations: FPT_FLS.1, FPT_FLS.2

327 None.

Audit: FPT_FLS.1

328 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Failures of the TSF, possibly with the type of the failure.

Audit: FPT_FLS.2

329 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Failures of the TSF, possibly with the type of the failure;

b) Whether the operations completed or whether the TOE entered a failure mode.

FPT_FLS.1 Fail secure

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_FLS.1.1 **The TSF shall, when the following types of failures occur: [assignment: *list of types of failures in the TSF*].**

FPT_FLS.1.2 **In failure mode, the following SFRs are no longer met: [assignment: *list of SFRs in the TSP*].**

FPT_FLS.1.3 **In failure mode, the following additional SFRs hold: [assignment: *list of SFRs*].**

FPT_FLS.2 Successful completion or fail secure

Hierarchical to: FPT_FLS.1 Fail secure

Dependencies: No dependencies.

FPT_FLS.2.1 The TSF shall when the following types of failures occur: [assignment: *list of types of failures in the TSF*] **either ensure that [assignment: *operations*] complete successfully or enter failure mode.**

FPT_FLS.2.2 In failure mode, the following SFRs are no longer met: [assignment: *list of SFRs in the TSP*].

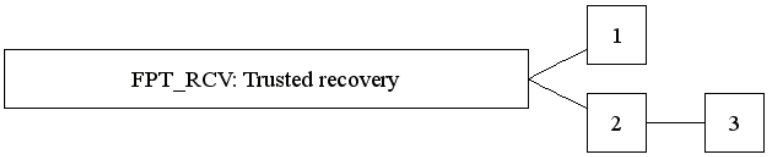
FPT_FLS.2.3 In failure mode, the following SFRs hold: [assignment: *list of SFRs*].

13.5 Trusted recovery (FPT_RCV)

330 The family provides requirements to ensure that the TSF may recover from failure mode to non-failure mode. See the Fail secure (FPT_FLS) family for an explanation of these two modes.

331 The conditions under which the TSF may enter failure mode should be specified by components from the FPT_FLS family. The PP/ST author should take special care to align the completion of the operations in these components and in the Trusted recovery (FPT_RCV) components.

Component levelling



Associated operations: FPT_RCV.1, FPT_RCV.2

332 None.

Associated operations: FPT_RCV.3

333 When FDP_ACC.1 or FDP_ACC.2 is included in the PP/ST, the PP/ST author may consider including the following operations:

a) Store_List_Of_Lost_Objects: an operation that allows a subject to store a list of objects that it has determined to be lost.

b) Store_List_Of_Remaining_Objects: an operation that allows a subject to store a list of objects that it has determined not to be lost

Audit: FPT_RCV.1

334 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The return of the TSF to normal mode, possibly with an identifier of the user or subject that returned it to normal mode.

Audit: FPT_RCV.2

335 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The return of the TSF to normal mode.

Audit: FPT_RCV.3

336 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) The return of the TSF to normal mode;

b) How many objects were lost;

c) Specific objects that were lost.

FPT_RCV.1 Manual recovery

Hierarchical to: No other components.

Dependencies: FPT_FLS.1 Fail secure

FPT_RCV.1.1 **After the TSF has entered failure mode due to [assignment: *list of types of failures in the TSF*], the TSF shall allow [assignment: *subject or user*] to restore it to non-failure mode.**

FPT_RCV.2 Automated recovery

Hierarchical to: No other components.

Dependencies: FPT_FLS.1 Fail secure

FPT_RCV.2.1 **After the TSF has entered failure mode due to [assignment: *list of types of failures in the TSF*], the TSF shall recover to non-failure mode using automated procedures.**

FPT_RCV.3 Automated recovery without undue loss

Hierarchical to: FPT_RCV.2 Automated recovery

Dependencies: FPT_FLS.1 Fail secure

FPT_RCV.3.1 After the TSF has entered failure mode due to [assignment: *list of types of failures in the TSF*], the TSF shall recover to non-failure mode using automated procedures.

FPT_RCV.3.2 **These automated procedures will not lose more than [assignment: *quantification*] of [assignment: *list of objects*].**

FPT_RCV.3.3 **The TSF shall allow [assignment: *subject*] to determine the objects that [selection: *were, were not*] lost.**

13.6 TSF physical protection (FPT_PHP)

337 TSF physical protection components refer to restrictions on unauthorised access to the TSF, and to the deterrence of, and resistance to, unauthorised physical modification, or substitution of the TSF.

338 Without SFRs based on these components, a TSF will lose its effectiveness in environments where physical tampering is unpreventable. This family also provides requirements regarding how the TSF shall respond to physical tampering attempts.

339 This family consists of four components:

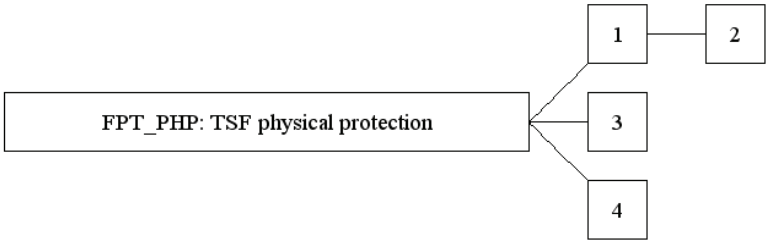
a) FPT_PHP.1 Tamper evidence which should be used to specify that physical attacks leave visible evidence. An example would be that the TOE is sealed with a seal that fragments when broken. FPT_PHP.1 Tamper evidence is targeted mainly to human users and may make no sense when applied to non-human users. Note that this "examining the visible evidence" by a user does not require the user to bind to a subject.

b) FPT_PHP.2 Tamper detection which should be used to specify that subjects can determine that physical attacks are occurring or have occurred. Where FPT_PHP.1 Tamper evidence requires physical inspection of a user, FPT_PHP.2 Tamper detection allows IT inspection, e.g. by a subject controlled by an administrator logging in every day or so, or an autonomous process.

c) FPT_PHP.3 Tamper response which should be used to specify that the TSF can detect and actively respond to physical attacks, by e.g. zeroising critical data or warning an administrator. Where FPT_PHP.1 Tamper evidence and FPT_PHP.2 Tamper detection are passive in nature (they allow detection), FPT_PHP.3 Tamper response is active: when detected, the TSF shall undertake action.

d) FPT_PHP.4 Resistance to physical attack which should be used to specify that the TSF can passively resist physical attacks. Examples include encasing the TOE in a strong shell or scrambling the implementation of an integrated circuit TOE that attackers do not know where to attack the TOE.

Component levelling



Associated operations: FPT_PHP.1, FPT_PHP.2, FPT_PHP.3, FPT_PHP.4

340 None.

Audit: FPT_PHP.1, FPT_PHP.4

341 There are no auditable events foreseen.

Audit: FPT_PHP.2

342 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful physical tampering scenarios.

Audit: FPT_PHP.3

343 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Successful/unsuccessful physical tampering scenarios;

b) Actions undertaken by the TSF.

FPT_PHP.1 Tamper evidence

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PHP.1.1 **The TSF shall show physical evidence to users when [assignment: *list of physical tampering scenarios*] to [selection: *the TSF, [assignment: list of TSF parts]*] have occurred.**

FPT_PHP.2 Tamper detection

Hierarchical to: FPT_PHP.1 Tamper evidence

Dependencies: No dependencies.

FPT_PHP.2.1 The TSF shall **detect** [assignment: *list of physical tampering scenarios*] to [selection: ***the*** *TSF, [assignment: list of TSF parts]*].

FPT_PHP.2.2 **The TSF shall allow [assignment: *subjects*] to determine whether these scenarios have occurred.**

FPT_PHP.3 Tamper response

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PHP.3.1 **The TSF shall detect [assignment: *list of physical tampering scenarios*] to [selection: *the TSF, [assignment: list of TSF parts]*].**

FPT_PHP.3.2 **The TSF shall respond to these scenarios by [assignment: *actions to be undertaken by the TSF*].**

FPT_PHP.4 Resistance to physical attack

Hierarchical to: No other components.

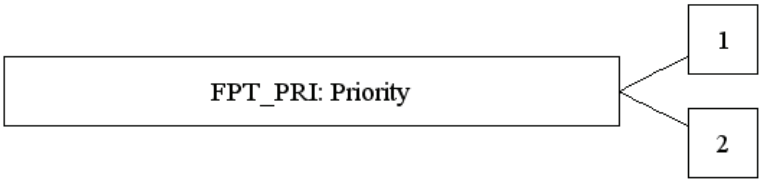
Dependencies: No dependencies.

FPT_PHP.4.1 **The TSF shall continue to meet the TSP when [assignment: *list of physical tampering scenarios*] to [selection: *the TSF, [assignment: list of TSF parts]*] occur.**

13.7 Priority (FPT_PRI)

344 The requirements of this family allow the TSF to control operations performed by subjects such that high priority subjects or high priority operations will be accomplished without undue interference or delay caused by low priority subjects or operations.

Component levelling



Associated operations: FPT_PRI.1, FPT_PRI.2

345 None.

Audit: FPT_PRI.1

346 There are no auditable events foreseen.

Audit: FPT_PRI.2

347 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) Operations that were performed later or not at all due to prioritisation, possibly with the subject attempting them.

FPT_PRI.1 Priority of subjects

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PRI.1.1 **When resources become overtaxed, the TSF shall ensure that subjects performing [assignment: *list of operations*] are prioritised using [assignment: *rules based on security attributes of subjects*].**

FPT_PRI.2 Priority of operations

Hierarchical to: No other components.

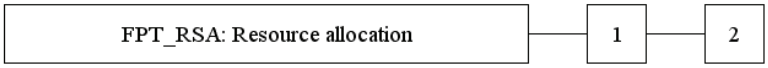
Dependencies: No dependencies.

FPT_PRI.2.1 **When resources become overtaxed, the TSF shall ensure that operations are prioritised using [assignment: *rules*].**

13.8 Resource allocation (FPT_RSA)

348 The requirements of this family allow the TSF to control the use of resources by subjects or objects such that denial of service will not occur because of unauthorised monopolisation of resources.

Component levelling



Associated operations: FPT_RSA.1, FPT_RSA.2

349 None.

Audit: FPT_RSA.1

350 There are no auditable events foreseen.

Audit: FPT_RSA.2

351 When FAU_GEN.1 or FAU_GEN.2 is included in the PP/ST, the PP/ST may consider auditing the following events:

a) A quotum being (almost) surpassed, with the object/subject (almost) surpassing its quotum;

b) Action(s) undertaken.

FPT_RSA.1 Maximum quotas for subjects and objects

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_RSA.1.1 **The TSF shall enforce maximum quotas for [selection: *processing resources, storage resources, communication resources, [assignment: other resources]*] that [assignment: *list of subjects and/or objects*] can use [selection: *simultaneously, over a specified period of time*].**

FPT_RSA.1.2 **The TSF shall [assignment: *action(s)*] when a maximum quotum is [selection: *almost surpassed, surpassed*].**

FPT_RSA.2 Minimum and maximum quotas for subjects and objects

Hierarchical to: FPT_RSA.1 Maximum quotas for subjects and objects

Dependencies: No dependencies.

FPT_RSA.2.1 The TSF shall enforce maximum quotas for [selection: *processing resources, storage resources, communication resources, [assignment: other resources]*] that [assignment: *list of subjects and/or objects*] can use [selection: *simultaneously, over a specified period of time*].

FPT_RSA.2.2 The TSF shall **ensure the provision of a minimum quantity of [selection: *processing resources, storage resources, communication resources,* *[assignment: other resources]*] that [assignment: *list of subjects and/or objects*] can use [selection: *simultaneously, over a specified period of time*].**

FPT_RSA.2.3 **The TSF shall [assignment: *action(s)*] when a maximum quotum is [selection: *almost surpassed, surpassed*].**

13.9 Residual information protection (FPT_RIP)

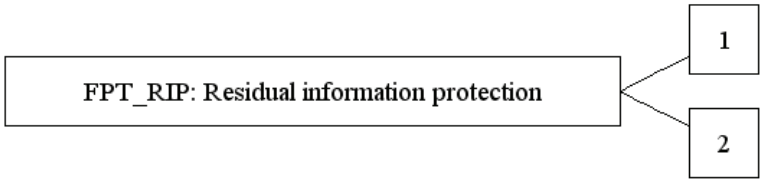
352 This family allows specification of permanent deletion of data from objects and subjects, to ensure that this data can never be retrieved.

353 The family has two components

a) FPT_RIP.1 Removal before use if a new object or subject is created, all existing information in that object/subject is irretrievably removed.

b) FPT_RIP.2 Removal after use certain operations on an object will cause all information in that object to be irretrievably removed;

Component levelling



Associated operations: FPT_RIP.1, FPT_RIP.2

354 None.

Audit: FPT_RIP.1, FPT_RIP.2

355 There are no auditable events foreseen.

FPT_RIP.1 Removal before use

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_RIP.1.1 **The TSF shall ensure that when [assignment: *list of objects and/or subjects*] is created, all information is irretrievably removed from those objects/subjects before any operation or other action is performed on them.**

FPT_RIP.2 Removal after use

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_RIP.2.1 **The TSF shall ensure that if [assignment: *list of operations*] are performed on [assignment: *list of objects*] the information in those objects is irretrievably removed.**

14 Class FMI: Miscellaneous

356 This class contains families that could not be fitted easily in other classes. It currently consists of three families: Random number generation (FMI_RND), Time stamps (FMI_TIM) and Choice (FMI_CHO).

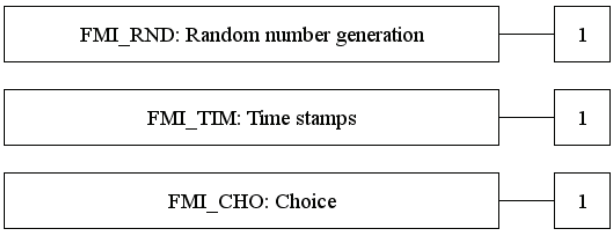


Figure 18 - FMI: Miscellaneous class decomposition

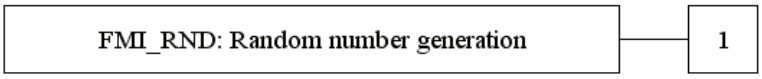
14.1 Random number generation (FMI_RND)

357 This family provides requirements for the generation of random numbers.

358 An example of a random number metric is “Shannon entropy of at least five bits”.

359 The dependency to FDP_ACC.1 Access control is included to show that access to the random number object(s) should be controlled.

Component levelling



Associated operations: FMI_RND.1

360 None.

Audit: FMI_RND.1

361 There are no auditable events foreseen.

FMI_RND.1 Random number generation

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FMI_RND.1.1 The TSF shall generate random numbers that meet [assignment: *metric(s)*].

FMI_RND.1.2 The TSF shall store these random numbers in [assignment: *object(s)*].

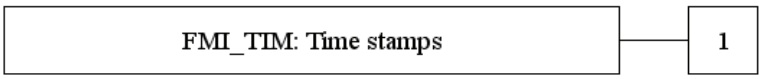
14.2 Time stamps (FMI_TIM)

362 This family addresses requirements for a time stamp function.

363 An example of an accuracy metric is “1/10th of a second”.

364 The dependency to FDP_ACC.1 Access control is included to show that access to the time object should be controlled.

Component levelling



Associated operations: FMI_TIM.1

365 None.

Audit: FMI_TIM.1

366 There are no auditable events foreseen.

FMI_TIM.1 Time stamps

Hierarchical to: No other components.

Dependencies: FDP_ACC.1 Access control

FMI_TIM.1.1 **The TSF shall maintain the current time in [assignment: *object*] to an accuracy of [assignment: *accuracy metric*].**

14.3 Choice (FMI_CHO)

367 This family allows a PP writer to specify that certain security objectives may be met by a variety of SFRs. This allows a more flexible specification of Protection Profiles.

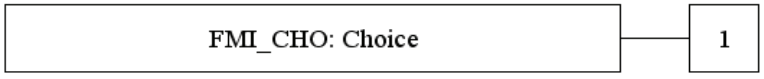
368 An example of the use of FMI_CHO.1 Choice is where a PP author wishes to specify residual information protection, does not care whether information is deleted before or after use, but does not wish to restrict his PP to either. In this case the PP author can use FMI_CHO.1 Choice to specify that either FPT_RIP.1 Removal before use or FPT_RIP.2 Removal after use is used.

369 FMI_CHO.1 Choice shall only be used in Protection Profiles. It is not allowed to use FMI_CHO.1 Choice in a Security Target. For the purpose of PP compliance, if a PP or ST contains the SFRs that are used in one of the options of FMI_CHO.1 Choice, the PP/ST contains FMI_CHO.1 Choice.

370 In the Residual information protection (FPT_RIP) example above, if the PP contained FMI_CHO.1 Choice specifying FPT_RIP.1 Removal before use or FPT_RIP.2 Removal after use, an ST with FPT_RIP.1 Removal before use would contain FMI_CHO.1 Choice for the purpose of conforming to that PP.

371 The PP author should consider that liberal and/or nested use of FMI_CHO.1 Choice can lead to PPs of great complexity.

Component levelling



Associated operations: FMI_CHO.1

372 None.

Audit: FMI_CHO.1

373 There are no auditable events foreseen.

FMI_CHO.1 Choice

Hierarchical to: No other components.

Dependencies: No dependencies.

FMI_CHO.1.1 **[assignment: *set of SFRs*] or [assignment: *set of SFRs*].**

A Dependency tables (normative)

374 The following dependency tables for functional components show their direct, indirect and optional dependencies. Each of the components that is a dependency of some functional component is allocated a column. Each functional component is allocated a row. The value in the table cell indicate whether the column label component is directly required (indicated by a cross "X"), indirectly required (indicated by a dash "-"), or optionally required (indicated by an "o") by the row label component. An example of a component with optional dependencies is FAU_GEN.2 Audit data generation with time, which requires either FMI_TIM.1 Time stamps, FCO_ITC.1 Import without security attributes or FCO_IID.1 Integrity of imported data without recovery to be present. So if FMI_TIM.1 Time stamps is present, neither FCO_ITC.1 Import without security attributes or FCO_IID.1 Integrity of imported data without recovery are necessary and vice versa. If no character is presented, the component is not dependent upon another component.

	FAU_GEN.1	FAU_SAA.1	FCO_IID.1	FCO_ITC.1	FDP_ACC.1	FDP_ISA.1	FMI_TIM.1	FPT_RSA.1
FAU_ARP.1	-	X			-	-		-
FAU_GEN.1					X	-		X
FAU_GEN.2			O	O	X	-	O	X
FAU_SAA.1	X				-	-		-
FAU_SAA.2	X				-	-		-
FAU_SAA.3	X				-	-		-
FAU_SAA.4								

Table 1 Dependency table for Class FAU: Security audit

	FCO_IED.1	FCO_IID.1	FCO_IID.2	FCO_IID.3	FDP_ACC.1	FDP_ISA.1
FCO_AED.1						
FCO_CED.1						
FCO_CID.1						
FCO_ETC.1						
FCO_IED.1						
FCO_IED.2						
FCO_IID.1						
FCO_IID.2						
FCO_IID.3						
FCO_ITC.1						
FCO_ITC.2		O	O			
FCO_NRE.1	X				X	-
FCO_NRE.2	X					
FCO_NRI.1	X	O	O	O		
FCO_NRI.2		O	O	O	X	-
FCO_TED.1						
FCO_TID.1						
FCO_UNE.1						
FCO_UNE.2						

Table 2 Dependency table for Class FCO: Communication

	FDP_ACC.1	FDP_ISA.1
FDP_ACC.1	-	X
FDP_ACC.2	-	X
FDP_ISA.1	X	-
FDP_MSA.1	X	-
FDP_MSA.2	X	-
FDP_ROL.1	X	-
FDP_UNL.1		
FDP_UNL.2		
FDP_UNL.3		
FDP_UNO.1		
FDP_UNO.2		

Table 3 Dependency table for Class FDP: Data protection and privacy

	FDP_ACC.1	FDP_ISA.1	FIA_UAU.1	FIA_UAU.2	FIA_UID.2	FIA_URE.2	FIA_USB.1
FIA_AFL.1	-	-	X		-	-	-
FIA_LOB.1							X
FIA_LOB.2							X
FIA_QAD.1	-	-				X	X
FIA_QAD.2	-	-				X	X
FIA_SUA.1							X
FIA_TBR.1							X
FIA_TIN.1							X
FIA_TIN.2					X		-
FIA_TOB.1							X
FIA_TOB.2							X
FIA_UAU.1	-	-			X	X	-
FIA_UAU.2					X		-
FIA_UAU.3	-	-	X		-	-	-
FIA_UAU.4	-	-	X		-	-	-
FIA_UAU.5	-	-	O	O	-	-	-
FIA_UAU.6	-	-	X		-	-	-
FIA_UID.1							X
FIA_UID.2							X
FIA_URE.1	X	-					
FIA_URE.2	X	-					
FIA_USB.1							

Table 4 Dependency table for Class FIA: Identification, Authentication and Binding

	FDP_ACC.1	FDP_ISA.1
FMI_CHO.1		
FMI_RND.1	X	-
FMI_TIM.1	X	-

Table 5 Dependency table for Class FMI: Miscellaneous

	FPT_FLS.1
FPT_FLS.1	
FPT_FLS.2	
FPT_FLT.1	
FPT_PHP.1	
FPT_PHP.2	
FPT_PHP.3	
FPT_PHP.4	
FPT_PRI.1	
FPT_PRI.2	
FPT_RCV.1	X
FPT_RCV.2	X
FPT_RCV.3	X
FPT_RIP.1	
FPT_RIP.2	
FPT_RSA.1	
FPT_RSA.2	
FPT_TOU.1	
FPT_TST.1	
FPT_TST.2	

Table 6 Dependency table for Class FPT: Protection of the TSF