Common Criteria

Common Methodology for Information Technology Security Evaluation

CEM-99/045

Part 2: Evaluation Methodology

Annotated with interpretations

as of 2003-12-31

Version 1.0

August 1999

Foreword

This document, version 1.0 of the Common Methodology for Information Technology Security Evaluation (CEM), is issued for use by the international IT security evaluation community. The CEM is a companion document to the Common Criteria for Information Technology Security Evaluation (CC) and is the result of extensive international cooperation. Practical experience acquired through use in evaluations, and requests for interpretations received, will be used to further develop the CEM.

A template for reporting observations on the CEM is included at the end of this document. Any observation reports should be communicated to one or more of the following points of contact at the sponsoring organisations:

CANADA:
Communications Security Establishment
Canadian Common Criteria Evaluation and Certification Scheme
P.O. Box 9703, Terminal
Ottawa, Ontario, Canada K1G 3Z4
Tel: +1.613.991.7543, Fax: +1.613.991.7455
E-mail: criteria@cse-cst.gc.ca
WWW: http://www.cse-cst.gc.ca/cse/english/cc.html

FRANCE:
Service Central de la Sécurité des Systèmes d'Information (SCSSI)
Centre de Certification de la Sécurité des Technologies de l'Information
18, rue du docteur Zamenhof
F-92131 Issy les Moulineaux, France
Tel: +33.1.41463784, Fax: +33.1.41463701
E-mail: ssi20@calva.net
WWW: http://www.scssi.gouv.fr

GERMANY:
Bundesamt für Sicherheit in der Informationstechnik (BSI)
Abteilung V
Postfach 20 03 63
D-53133 Bonn, Germany
Tel: +49.228.9582.300, Fax: +49.228.9582.427
E-mail: cc@bsi.de
WWW: http://www.bsi.de/cc

NETHERLANDS:
Netherlands National Communications Security Agency
P.O. Box 20061
NL 2500 EB The Hague
The Netherlands
Tel: +31.70.3485637, Fax: +31.70.3486503
E-mail: criteria@nlncsa.minbuza.nl
WWW: http://www.tno.nl/instit/fel/refs/cc.html

UNITED KINGDOM:
Communications-Electronics Security Group
Compusec Evaluation Methodology
P.O. Box 144
Cheltenham GL52 5UE, United Kingdom
Tel: +44.1242.221.491 ext. 5257, Fax: +44.1242.252.291
E-mail: criteria@cesg.gov.uk
WWW: http://www.cesg.gov.uk/cchtml
FTP: ftp://ftp.cesg.gov.uk/pub

UNITED STATES - NIST:
National Institute of Standards and Technology
Computer Security Division
100 Bureau Drive, MS: 8930
Gaithersburg, Maryland 20899, U.S.A.
Tel: +1.301.975.5390, Fax: +1.301.948.0279
E-mail: criteria@nist.gov
WWW: http://csrc.nist.gov/cc

UNITED STATES - NSA:
National Security Agency
Attn: V1, Common Criteria Technical Advisor
Fort George G. Meade, Maryland 20755-6740, U.S.A.
Tel: +1.410.854.4458, Fax: +1.410.854.7512
E-mail: common_criteria@radium.ncsc.mil
WWW: http://www.radium.ncsc.mil/tpep/

Version 1.0

August 1999

-

Preface to this Annotated Version

This version of the Common Methodology for Information Technology Security Evaluation Part 2 (CEM) v1.0 incorporates all Final Interpretations that have been approved by the Common Criteria Interpretations Management Board (CCIMB). Those interpretations that affect the text of the CEM v1.0 are listed in the table below, together with their effective dates, and the portions of this document that are changed as a result of their approval. In addition, the methodology created for Flaw Remediation (ALC_FLR), though not a part of any EAL, has been included at the end of this document.

Within this version of the CEM v1.0, each of the affected areas of text is preceded by an Interpretation Note that identifies the Interpretation that resulted in the change. All changes to text are indicated in red: added text is underscored and deleted text is struck-through.

In order to minimise confusion when comparing the original version 1.0 and this annotated version, paragraphs, components, etc have *not* been renumbered -- deleted paragraphs are simply removed; new paragraphs have been inserted without numbers. This approach maintains the original numbering.

The complete list of all Final Interpretations approved by the CCIMB can be found at http://www.commoncriteria.org/ri/FinalRI/Final_Interpretations.html

Number	Effective Date	CEM v1.0 References
003	11 Feb 2002	2:ACM_CAP.2, 3:ACM_CAP.3, 4:ACM_CAP.4
004	12 Nov 2001	3:ACM_SCP.1, 4:ACM_SCP.2
008	31 July 2001	ASE_INT
016	11 Feb 2002	2:ADO_DEL.1, 3:ADO_DEL.1, 4:ADO_DEL.2
019	11 Feb 2002	ASE_REQ, APE_REQ
024	16 Feb 2001	Section 2.2.2
025	31 July 2001	Annex B.6 (“TOE Boundary”)
031 rev1	25 Oct 2002	2:AVA_VLA.1, 3:AVA_VLA.1, 4:AVA_VLA.2
038	31 Oct 2003	3:ACM_SCP.1, 4:ACM_SCP.2
049	16 Feb 2001	ASE_OBJ, APE_OBJ
051 rev1	25 Oct 2002	2:AVA_VLA.1, 3:AVA_VLA.1, 4:AVA_VLA.2
064	16 Feb 2001	APE_SRE, ASE_SRE
069	30 Mar 2001	4:ADV_SPM.1
074	15 Oct 2000	3:ATE_COV.2, 3:ATE_DPT.1, 4:ATE_COV.2, 4:ATE_DPT.1
075	15 Oct 2000	1:ATE_IND.1, 2:ATE_FUN.1, 2:ATE_IND.2, 3:ATE_FUN.1, 3:ATE_IND.2, 4:ATE_FUN.1, 4:ATE_IND.2
080	15 Oct 2000	APE_REQ
084	16 Feb 2001	APE_REQ, ASE_REQ
085	11 Feb 2002	APE_REQ, ASE_REQ
116	31 July 2001	2:ADO_DEL.1, 3:ADO_DEL.1, 4:ADO_DEL.2
120	12 Nov 2001	Annex B.2 (“Sampling”)
127 rev1	25 Oct 2002	ASE_TSS
128 rev1	15 Nov 2002	4:ADO_DEL.2
133 rev1	25 Oct 2002	ASE_ENV
138	05 June 2002	ASE_REQ, APE_REQ

-

Number	Effective Date	CEM v1.0 References
222	31 Oct 2003	Section 1.3.2 (“Verb Usage”)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Scope

1 The Common Methodology for Information Technology Security Evaluation (CEM) is a companion document to the Common Criteria for Information Technology Security Evaluation (CC). The CEM describes the minimum actions to be performed by an evaluator in order to conduct a CC evaluation, using the criteria and evaluation evidence defined in the CC.

2 The scope of this version is limited to evaluations of Protection Profiles and TOEs for EAL1 through EAL4, as defined in the CC. It does not provide guidance for EALs 5 through 7, nor for evaluations using other assurance packages. The CEM is based on CC version 2.1, including feedback resulting from interaction with the CC Interpretations Management Board (CCIMB).

3 The target audience for the CEM is primarily evaluators applying the CC and certifiers confirming evaluator actions; evaluation sponsors, developers, PP/ST authors and other parties interested in IT security may be a secondary audience.

4 The CEM recognises that not all questions concerning IT security evaluation will be answered herein and that further interpretations will be needed. Individual schemes will determine how to handle such interpretations, although these may be subject to mutual recognition agreements. A list of methodology-related activities that may be handled by individual schemes can be found in Annex B.9.

5 The CEM Part 1, v0.6 defined the general model for the CEM but is currently undergoing revision. Therefore, CEM Part 2 material takes precedence over any seemingly contradictory material with CEM Part 1. Future versions of Part 1 will resolve any such contradictions.

1.2 Organisation

6 This part, *CEM Part 2*, is divided into the following chapters:

7 Chapter 1, *Introduction*, describes the objectives, organisation, document conventions and terminology, and evaluator verdicts.

8 Chapter 2, *General evaluation tasks*, describes the tasks that are relevant for all evaluation activities. These are the tasks used to manage the inputs and prepare the outputs.

9 Chapter 3, *PP evaluation*, describes the methodology for the evaluation of Protection Profiles, based on the APE class of CC Part 3.

10 Chapter 4, *ST evaluation*, describes the methodology for the evaluation of Security Targets, based on the ASE class of CC Part 3.

11 Chapters 5 through 8 describe the evaluation methodology for the Evaluation Assurance Levels EAL1 to EAL4 defined in CC Part 3.

12 Annex A, *Glossary*, defines vocabulary and references used in the CEM and presents abbreviations and acronyms.

13 Annex B, *General evaluation guidance*, provides guidance common to several activities described in Chapters 3 through 8.

14 Annex C, *Providing CEM observation reports*, provides the CEM observation report guidance, example observations, and a template to be used for observation reports.

1.3 Document conventions

1.3.1 Terminology

15 The glossary, presented in Annex A of this part, includes only those terms used in a specialised way within this document. The majority of terms are used according to their accepted definitions.

16 The term *activity* is used to describe the application of an assurance class of the CC Part 3.

17 The term *sub-activity* is used to describe the application of an assurance component of the CC Part 3. Assurance families are not explicitly addressed in the CEM because evaluations are conducted on a single assurance component from an assurance family.

18 The term *action* is related to an evaluator action element of the CC Part 3. These actions are either explicitly stated as evaluator actions or implicitly derived from developer actions (implied evaluator actions) within the CC Part 3 assurance components.

19 The term *work unit* is the most granular level of evaluation work. Each CEM action comprises one or more work units, which are grouped within the CEM action by CC content and presentation of evidence or developer action element. The work units are presented in the CEM in the same order as the CC elements from which they are derived. Work units are identified in the left margin by a symbol such as *4:ALC_TAT.1-2*. In this symbol, the first digit (*4*) indicates the EAL; the string *ALC_TAT.1* indicates the CC component (i.e. the CEM sub-activity), and the final digit (*2*) indicates that this is the second work unit in the ALC_TAT.1 sub-activity.

20 Unlike the CC, where each element maintains the last digit of its identifying symbol for all components within the family, the CEM may introduce new work

units when a CC evaluator action element changes from sub-activity to sub-activity; as a result, the last digit of the work unit's identifying symbol may change although the work unit remains unchanged. For example, because an additional work unit labeled 4:ADV_FSP.2-7 was added at EAL4, the subsequent sequential numbering of FSP work units is offset by one. Thus work unit 3:ADV_FSP.1-8 is now mirrored by work unit 4:ADV_FSP.2-9; each express the same requirement though their numbering no longer directly correspond.

21 Any methodology-specific evaluation work required that is not derived directly from CC requirements is termed *task* or *sub-task*.

1.3.2 Verb usage

22 All work unit and sub-task verbs are preceded by the auxiliary verb *shall* and by presenting both the verb and the *shall* in ***bold italic*** type face. The auxiliary verb *shall* is used only when the provided text is mandatory and therefore only within the work units and sub-tasks. The work units and sub-tasks contain mandatory activities that the evaluator must perform in order to assign verdicts.

Interp Note : The following paragraph is changed as a result of Interpretation 222.

23 Guidance text accompanying work units and sub-tasks gives further explanation on how to apply the CC words in an evaluation. The described method is normative, meaning that the verb usage is in accordance with ISO definitions for these verbs; that is: the ~~The~~ auxiliary verb *should* is used when the described method is strongly preferred~~, but others may be justifiable. The~~ and the auxiliary verb *may* is used where ~~something~~the described method(s) is allowed but no preference is indicated. (The auxiliary verb *shall* is used only for the text of work units.)

24 The verbs *check, examine, report* and *record* are used with a precise meaning within this part of the CEM and the glossary should be referenced for their definitions.

1.3.3 General evaluation guidance

25 Material that has applicability to more than one sub-activity is collected in one place. Guidance whose applicability is widespread (across activities and EALs) has been collected into Annex B. Guidance that pertains to multiple sub-activities within a single activity has been provided in the introduction to that activity. If guidance pertains to only a single sub-activity, it is presented within that sub-activity.

1.3.4 Relationship between CC and CEM structures

26 There are direct relationships between the CC structure (i.e. class, family, component and element) and the structure of the CEM. Figure 1.1 illustrates the correspondence between the CC constructs of class, component and evaluator action elements and CEM activities, sub-activities and actions. However, several

CEM work units may result from the requirements noted in CC developer action and content and presentation elements.

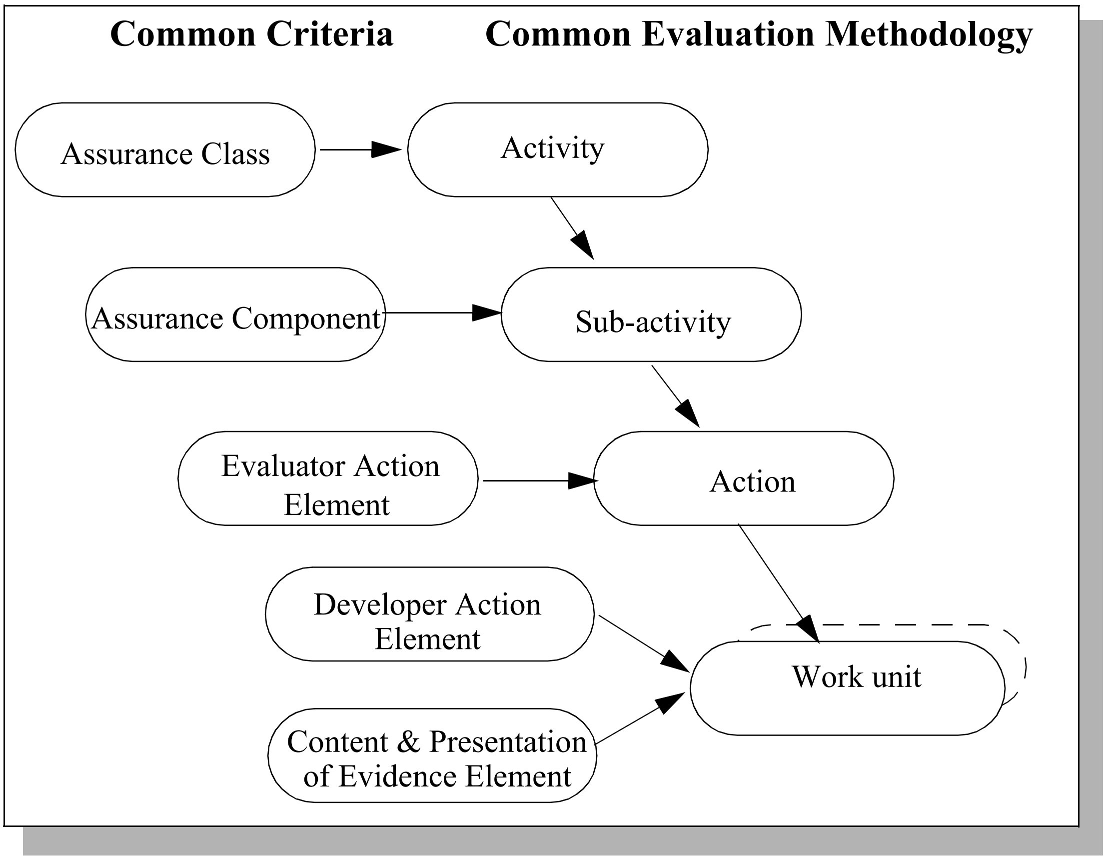

Figure 1.1 Mapping of the CC and CEM structures

1.4 Evaluator verdicts

27 The evaluator assigns verdicts to the requirements of the CC and not to those of the CEM. The most granular CC structure to which a verdict is assigned is the evaluator action element (explicit or implied). A verdict is assigned to an applicable CC evaluator action element as a result of performing the corresponding CEM action and its constituent work units. Finally, an evaluation result is assigned, as described in CC Part 1, Section 5.3.

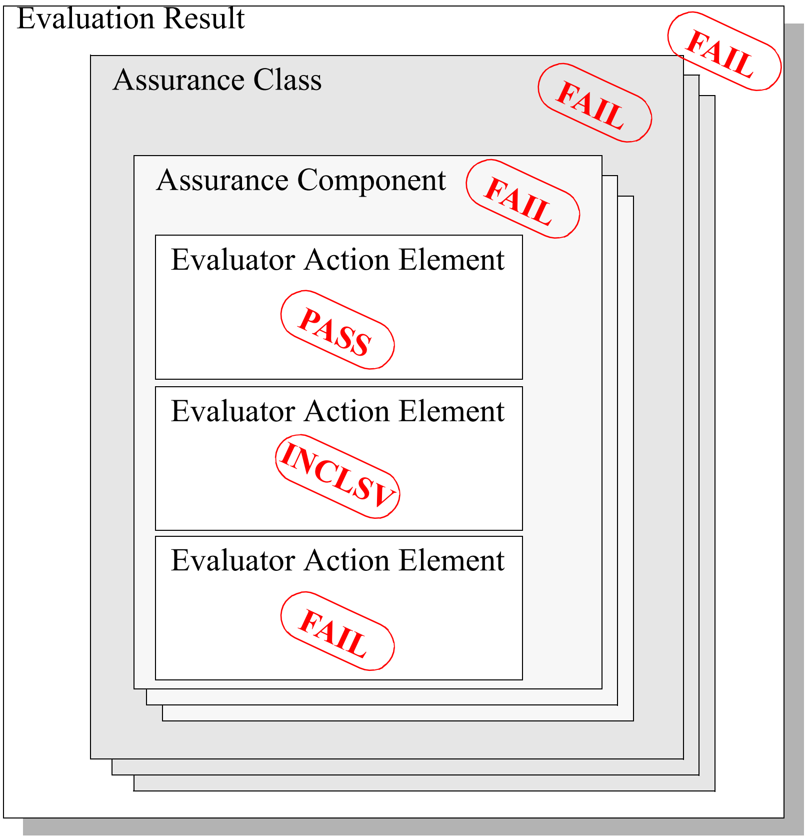

Figure 1.2 Example of the verdict assignment rule

28 The CEM recognises three mutually exclusive verdict states:

a) Conditions for a *pass* verdict are defined as an evaluator completion of the CC evaluator action element and determination that the requirements for the PP, ST or TOE under evaluation are met. The conditions for passing the element are defined as the constituent work units of the related CEM action;

b) Conditions for an *inconclusive* verdict are defined as an evaluator incompletion of one or more work units of the CEM action related to the CC evaluator action element;

c) Conditions for a *fail* verdict are defined as an evaluator completion of the CC evaluator action element and determination that the requirements for the PP, ST, or TOE under evaluation are not met.

29 All verdicts are initially *inconclusive* and remain so until either a *pass* or *fail* verdict is assigned.

30 The overall verdict is *pass* if and only if all the constituent verdicts are also *pass*. In the example illustrated in Figure 1.2, if the verdict for one evaluator action element is *fail* then the verdicts for the corresponding assurance component, assurance class, and overall verdict are also *fail.*

Chapter 2

General evaluation tasks

2.1 Introduction

31 All evaluations, whether of a PP or TOE (including ST), have two evaluator tasks in common: the input task and the output task. These two tasks, which are related to management of evaluation evidence and to report generation, are described in this chapter. Each task has associated sub-tasks that apply to, and are normative for all CC evaluations (evaluation of a PP or a TOE).

32 Although the CC does not mandate specific requirements on these evaluation tasks, the CEM does so where it is necessary to ensure conformance with the universal principles defined in Part 1 of the CEM. In contrast to the activities described elsewhere in this part of the CEM, these tasks have no verdicts associated with them as they do not map to CC evaluator action elements; they are performed in order to comply with the CEM.

2.2 Evaluation input task

2.2.1 Objectives

33 The objective of this task is to ensure that the evaluator has available the correct version of the evaluation evidence necessary for the evaluation and that it is adequately protected. Otherwise, the technical accuracy of the evaluation cannot be assured, nor can it be assured that the evaluation is being conducted in a way to provide repeatable and reproducible results.

2.2.2 Application notes

Interp Note : The following paragraph is changed as a result of Interpretation 024.

34 The responsibility to provide all the required evaluation evidence lies with the sponsor. However, most of the evaluation evidence is likely to be produced and supplied by the developer, on behalf of the sponsor. Since the assurance requirements apply to the entire TOE, evaluation evidence pertaining to all products that are part of the TOE is made available to the evaluator. The scope and required content of such evaluation evidence is independent of the level of control that the developer has over each of the products that are part of the TOE. For example, if a high-level design is required, then the ADV_HLD requirements will apply to all subsystems that are part of the TSF. In addition, assurance requirements that call for procedures to be in place (for example, ACM_CAP and ADO_DEL) will also apply to the entire TOE (including any product from another developer).

35 It is recommended that the evaluator, in conjunction with the sponsor, produce an index to required evaluation evidence. This index may be a set of references to the documentation. This index should contain enough information (e.g. a brief summary of each document, or at least an explicit title, indication of the sections of interest) to help the evaluator to find easily the required evidence.

36 It is the information contained in the evaluation evidence that is required, not any particular document structure. Evaluation evidence for a sub-activity may be provided by separate documents, or a single document may satisfy several of the input requirements of a sub-activity.

37 The evaluator requires stable and formally-issued versions of evaluation evidence. However, draft evaluation evidence may be provided during an evaluation, for example, to help an evaluator make an early, informal assessment, but is not used as the basis for verdicts. It may be helpful for the evaluator to see draft versions of particular appropriate evaluation evidence, such as:

a) test documentation, to allow the evaluator to make an early assessment of tests and test procedures;

b) design documents, to provide the evaluator with background for understanding the TOE design;

c) source code or hardware drawings, to allow the evaluator to assess the application of the developer's standards.

38 Draft evaluation evidence is more likely to be encountered where the evaluation of a TOE is performed concurrently with its development. However, it may also be encountered during the evaluation of an already-developed TOE where the developer has had to perform additional work to address a problem identified by the evaluator (e.g. to correct an error in design or implementation) or to provide evaluation evidence of security that is not provided in the existing documentation (e.g. in the case of a TOE not originally developed to meet the requirements of the CC).

2.2.3 Management of evaluation evidence sub-task

2.2.3.1 Configuration control

39 The evaluator ***shall perform*** configuration control of the evaluation evidence.

40 The CC implies that the evaluator is able to identify and locate each item of evaluation evidence after it has been received and is able to determine whether a specific version of a document is in the evaluator’s possession.

41 The evaluator ***shall protect*** the evaluation evidence from alteration or loss while it is in the evaluator’s possession.

2.2.3.2 Disposal

42 Schemes may wish to control the disposal of evaluation evidence at the conclusion of an evaluation. The disposal of the evaluation evidence should be achieved by one or more of:

a) returning the evaluation evidence;

b) archiving the evaluation evidence;

c) destroying the evaluation evidence.

2.2.3.3 Confidentiality

43 An evaluator may have access to sponsor and developer commercially-sensitive information (e.g. TOE design information, specialist tools), and may have access to nationally-sensitive information during the course of an evaluation. Schemes may wish to impose requirements for the evaluator to maintain the confidentiality of the evaluation evidence. The sponsor and evaluator may mutually agree to additional requirements as long as these are consistent with the scheme.

44 Confidentiality requirements affect many aspects of evaluation work, including the receipt, handling, storage and disposal of evaluation evidence.

2.3 Evaluation output task

2.3.1 Objectives

45 The objective of this section is to describe the Observation Report (OR) and the Evaluation Technical Report (ETR). Schemes may require additional evaluator reports such as reports on individual units of work, or may require additional information to be contained in the OR and the ETR. The CEM does not preclude the addition of information into these reports as the CEM specifies only the minimum information content.

46 Consistent reporting of evaluation results facilitates the achievement of the universal principle of repeatability and reproducibility of results. The consistency covers the type and the amount of information reported in the ETR and OR. ETR and OR consistency among different evaluations is the responsibility of the overseer.

47 The evaluator performs the two following sub-tasks in order to achieve the CEM requirements for the information content of reports:

a) write OR sub-task (if needed in the context of the evaluation);

b) write ETR sub-task.

2.3.2 Application notes

48 In this version of the CEM, the requirements for the provision of evaluator evidence to support re-evaluation and re-use have not been explicitly stated. The information resulting from evaluator work to assist in re-evaluation or re-use has not yet been determined. Where information for re-evaluation or re-use is required by the sponsor, the scheme under which the evaluation is being performed should be consulted.

2.3.3 Write OR sub-task

49 ORs provide the evaluator with a mechanism to request a clarification (e.g. from the overseer on the application of a requirement) or to identify a problem with an aspect of the evaluation.

50 In the case of a fail verdict, the evaluator ***shall provide*** an OR to reflect the evaluation result.

51 The evaluator may also use ORs as one way of expressing clarification needs.

52 For each OR, the evaluator ***shall report*** the following:

a) the identifier of the PP or TOE evaluated;

b) the evaluation task/sub-activity during which the observation was generated;

c) the observation;

d) the assessment of its severity (e.g. implies a fail verdict, holds up progress on the evaluation, requires a resolution prior to evaluation being completed);

e) the identification of the organisation responsible for resolving the issue;

f) the recommended timetable for resolution;

g) the assessment of the impact on the evaluation of failure to resolve the observation.

53 The intended audience of an OR and procedures for handling the report depend on the nature of the report's content and on the scheme. Schemes may distinguish different types of ORs or define additional types, with associated differences in required information and distribution (e.g. evaluation ORs to overseers and sponsors).

2.3.4 Write ETR sub-task

2.3.4.1 Objectives

54 The evaluator ***shall provide*** an ETR to present technical justification of the verdicts.

55 The ETR may contain information proprietary to the developer or the sponsor.

56 The CEM defines the ETR's minimum content requirement; however, schemes may specify additional content and specific presentational and structural requirements. For instance, schemes may require that certain introductory material (e.g. disclaimers, and copyright clauses) be reported in the ETR.

57 The reader of the ETR is assumed to be familiar with general concepts of information security, the CC, the CEM, evaluation approaches and IT.

58 The ETR supports the overseer in providing the oversight verdict, but it is anticipated that it may not provide all of the information needed for oversight, and the documented results may not provide the evidence necessary for the scheme to confirm that the evaluation was done to the required standard. This aspect is outside the scope of the CEM and should be met using other oversight methods.

2.3.4.2 ETR for a PP Evaluation

59 This section describes the minimum content of the ETR for a PP evaluation. The contents of the ETR are portrayed in Figure 2.1; this figure may be used as a guide when constructing the structural outline of the ETR document.

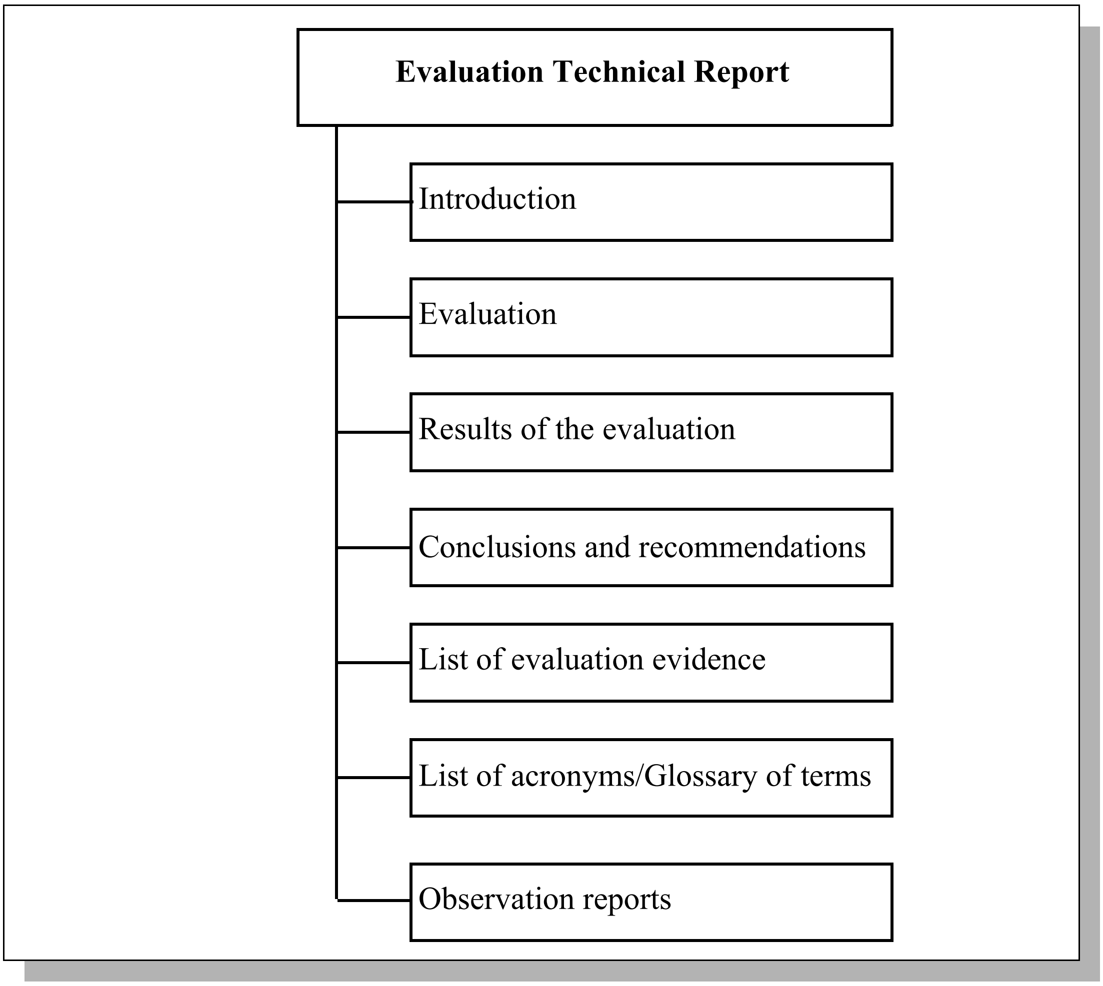

Figure 2.1 ETR information content for a PP evaluation

2.3.4.2.1 Introduction

60 The evaluator ***shall report*** evaluation scheme identifiers.

61 Evaluation scheme identifiers (e.g. logos) are the information required to unambiguously identify the scheme responsible for the evaluation oversight.

62 The evaluator ***shall report*** ETR configuration control identifiers.

63 The ETR configuration control identifiers contain information that identifies the ETR (e.g. name, date and version number).

64 The evaluator ***shall report*** PP configuration control identifiers.

65 PP configuration control identifiers (e.g. name, date and version number) are required to identify what is being evaluated in order for the overseer to verify that the verdicts have been assigned correctly by the evaluator.

66 The evaluator ***shall report*** the identity of the developer.

67 The identity of the PP developer is required to identify the party responsible for producing the PP.

68 The evaluator ***shall report*** the identity of the sponsor.

69 The identity of the sponsor is required to identify the party responsible for providing evaluation evidence to the evaluator.

70 The evaluator ***shall report*** the identity of the evaluator.

71 The identity of the evaluator is required to identify the party performing the evaluation and responsible for the evaluation verdicts.

2.3.4.2.2 Evaluation

72 The evaluator ***shall report*** the evaluation methods, techniques, tools and standards used.

73 The evaluator references the evaluation criteria, methodology and interpretations used to evaluate the PP.

74 The evaluator ***shall report*** any constraints on the evaluation, constraints on the handling of evaluation results and assumptions made during the evaluation that have an impact on the evaluation results.

75 The evaluator may include information in relation to legal or statutory aspects, organisation, confidentiality, etc.

2.3.4.2.3 Results of the evaluation

76 The evaluator ***shall report*** a verdict and a supporting rationale for each assurance component that constitutes an APE activity, as a result of performing the corresponding CEM action and its constituent work units.

77 The rationale justifies the verdict using the CC, the CEM, any interpretations and the evaluation evidence examined and shows how the evaluation evidence does or does not meet each aspect of the criteria. It contains a description of the work performed, the method used, and any derivation of results. The rationale may provide detail to the level of a CEM work unit.

2.3.4.2.4 Conclusions and recommendations

78 The evaluator ***shall report*** the conclusions of the evaluation, in particular the overall verdict as defined in CC Part 1 Chapter 5, and determined by application of the verdict assignment described in Section 1.4, Evaluator verdicts.

79 The evaluator provides recommendations that may be useful for the overseer. These recommendations may include shortcomings of the PP discovered during the evaluation or mention of features which are particularly useful.

2.3.4.2.5 List of evaluation evidence

80 The evaluator ***shall report*** for each item of evaluation evidence the following information:

- the issuing body (e.g. the developer, the sponsor);
- the title;
- the unique reference (e.g. issue date and version number).

2.3.4.2.6 List of acronyms/Glossary of terms

81 The evaluator ***shall report*** any acronyms or abbreviations used in the ETR.

82 Glossary definitions already defined by the CC or CEM need not be repeated in the ETR.

2.3.4.2.7 Observation reports

83 The evaluator ***shall report*** a complete list that uniquely identifies the ORs raised during the evaluation and their status.

84 For each OR, the list should contain its identifier as well as its title or a brief summary of its content.

2.3.4.3 ETR for a TOE Evaluation

85 This section describes the minimum content of the ETR for a TOE evaluation. The contents of the ETR are portrayed in Figure 2.2; this figure may be used as a guide when constructing the structural outline of the ETR document.

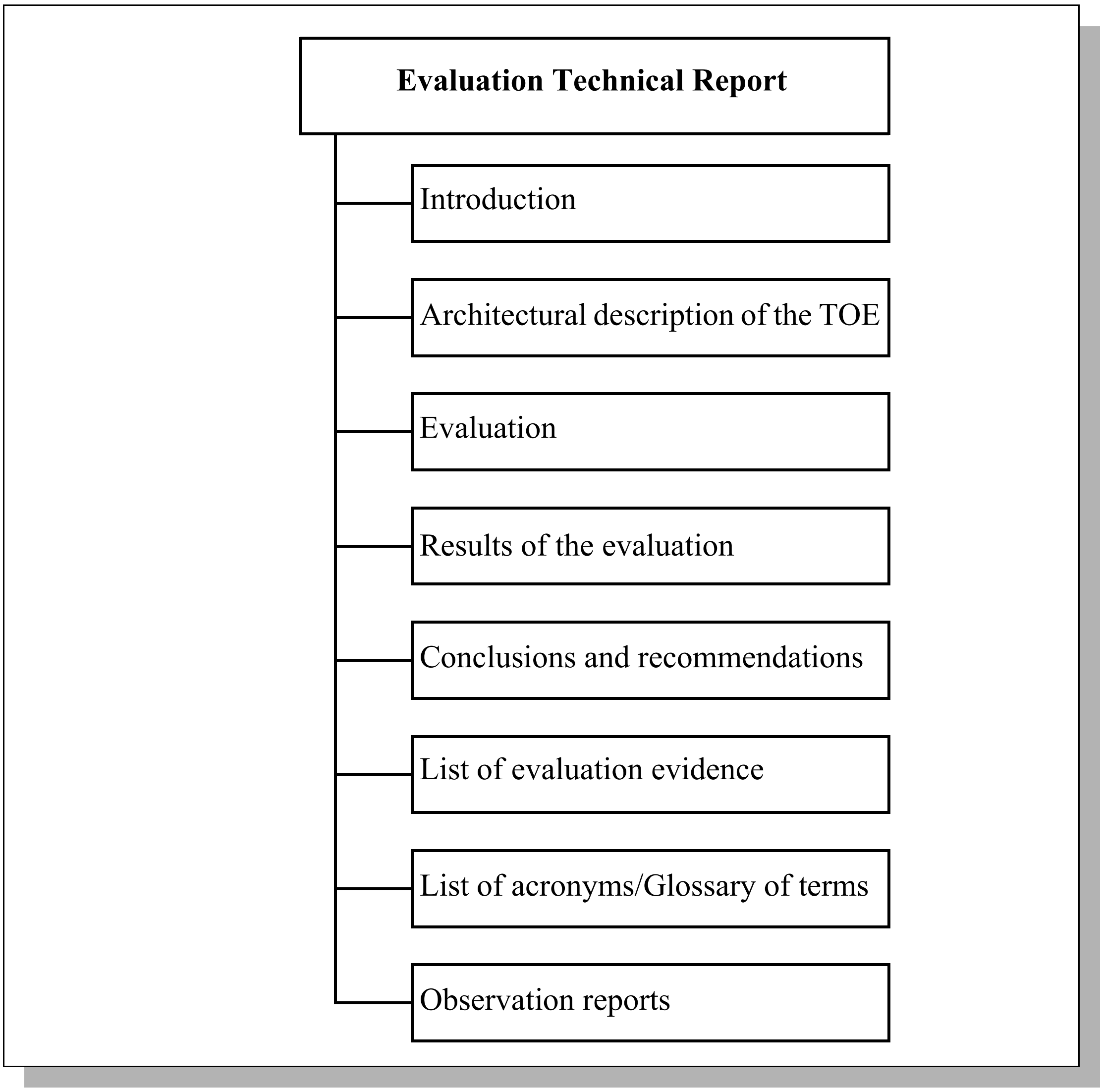

Figure 2.2 ETR information content for a TOE evaluation

2.3.4.3.1 Introduction

86 The evaluator ***shall report*** evaluation scheme identifiers.

87 Evaluation scheme identifiers (e.g. logos) are the information required to unambiguously identify the scheme responsible for the evaluation oversight.

88 The evaluator ***shall report*** ETR configuration control identifiers.

89 The ETR configuration control identifiers contain information that identifies the ETR (e.g. name, date and version number).

90 The evaluator ***shall report*** ST and TOE configuration control identifiers.

91 ST and TOE configuration control identifiers identify what is being evaluated in order for the overseer to verify that the verdicts have been assigned correctly by the evaluator.

92 If the ST claims that the TOE conforms with the requirements of one or more PPs, the ETR ***shall report*** the reference of the corresponding PPs.

93 The PPs reference contains information that uniquely identifies the PPs (e.g. title, date, and version number).

94 The evaluator ***shall report*** the identity of the developer.

95 The identity of the TOE developer is required to identify the party responsible for producing the TOE.

96 The evaluator ***shall report*** the identity of the sponsor.

97 The identity of the sponsor is required to identify the party responsible for providing evaluation evidence to the evaluator.

98 The evaluator ***shall report*** the identity of the evaluator.

99 The identity of the evaluator is required to identify the party performing the evaluation and responsible for the evaluation verdicts.

2.3.4.3.2 Architectural description of the TOE

100 The evaluator ***shall report*** a high-level description of the TOE and its major components based on the evaluation evidence described in the CC assurance family entitled “Development - high-level design (ADV_HLD)”, where applicable.

101 The intent of this section is to characterise the degree of architectural separation of the major components. If there is no high-level design (ADV_HLD) requirement in the ST, this is not applicable and is considered to be satisfied.

2.3.4.3.3 Evaluation

102 The evaluator ***shall report*** the evaluation methods, techniques, tools and standards used.

103 The evaluator may reference the evaluation criteria, methodology and interpretations used to evaluate the TOE or the devices used to perform the tests.

104 The evaluator ***shall report*** any constraints on the evaluation, constraints on the distribution of evaluation results and assumptions made during the evaluation that have an impact on the evaluation results.

105 The evaluator may include information in relation to legal or statutory aspects, organisation, confidentiality, etc.

2.3.4.3.4 Results of the evaluation

106 For each activity on which the TOE is evaluated, the evaluator ***shall report***:

- the title of the activity considered;

- a verdict and a supporting rationale for each assurance component that constitutes this activity, as a result of performing the corresponding CEM action and its constituent work units.

107 The rationale justifies the verdict using the CC, the CEM, any interpretations and the evaluation evidence examined and shows how the evaluation evidence does or does not meet each aspect of the criteria. It contains a description of the work performed, the method used, and any derivation of results. The rationale may provide detail to the level of a CEM work unit.

108 The evaluator ***shall report*** all information specifically required by a work unit.

109 For the AVA and ATE activities, work units that identify information to be reported in the ETR have been defined.

2.3.4.3.5 Conclusions and recommendations

110 The evaluator ***shall report*** the conclusions of the evaluation, which will relate to whether the TOE has satisfied its associated ST, in particular the overall verdict as defined in CC Part 1 Chapter 5, and determined by application of the verdict assignment described in Section 1.4, Evaluator verdicts.

111 The evaluator provides recommendations that may be useful for the overseer. These recommendations may include shortcomings of the IT product discovered during the evaluation or mention of features which are particularly useful.

2.3.4.3.6 List of evaluation evidence

112 The evaluator ***shall report*** for each item of evaluation evidence the following information:

- the issuing body (e.g. the developer, the sponsor);

- the title;

- the unique reference (e.g. issue date and version number).

2.3.4.3.7 List of acronyms/Glossary of terms

113 The evaluator ***shall report*** any acronyms or abbreviations used in the ETR.

114 Glossary definitions already defined by the CC or CEM need not be repeated in the ETR.

2.3.4.3.8 Observation reports

115 The evaluator ***shall report*** a complete list that uniquely identifies the ORs raised during the evaluation and their status.

116 For each OR, the list should contain its identifier as well as its title or a brief summary of its content.

2.4 Evaluation sub-activities

117 Figure 2.3 provides an overview of the work to be performed for an evaluation.

118 The evaluation evidence may vary depending upon the type of evaluation (PP evaluations require merely the PP, while TOE evaluations require TOE-specific evidence). Evaluation outputs result in an ETR and possibly ORs. The evaluation sub-activities vary and, in the case of TOE evaluations, depend upon the assurance requirements in the CC Part 3.

119 Each of the Chapters 3 through 8 is organised similarly based on the evaluation work required for an evaluation. Chapter 3 addresses the work necessary for reaching an evaluation result on a PP. Chapter 4 addresses the work necessary on an ST, although there is no separate evaluation result for this work. Chapters 5 through 8 address the work necessary for reaching an evaluation result on EAL1 through EAL4 (in combination with the ST). Each of these chapters is meant to stand alone and hence may contain some repetition of text that is included in other chapters.

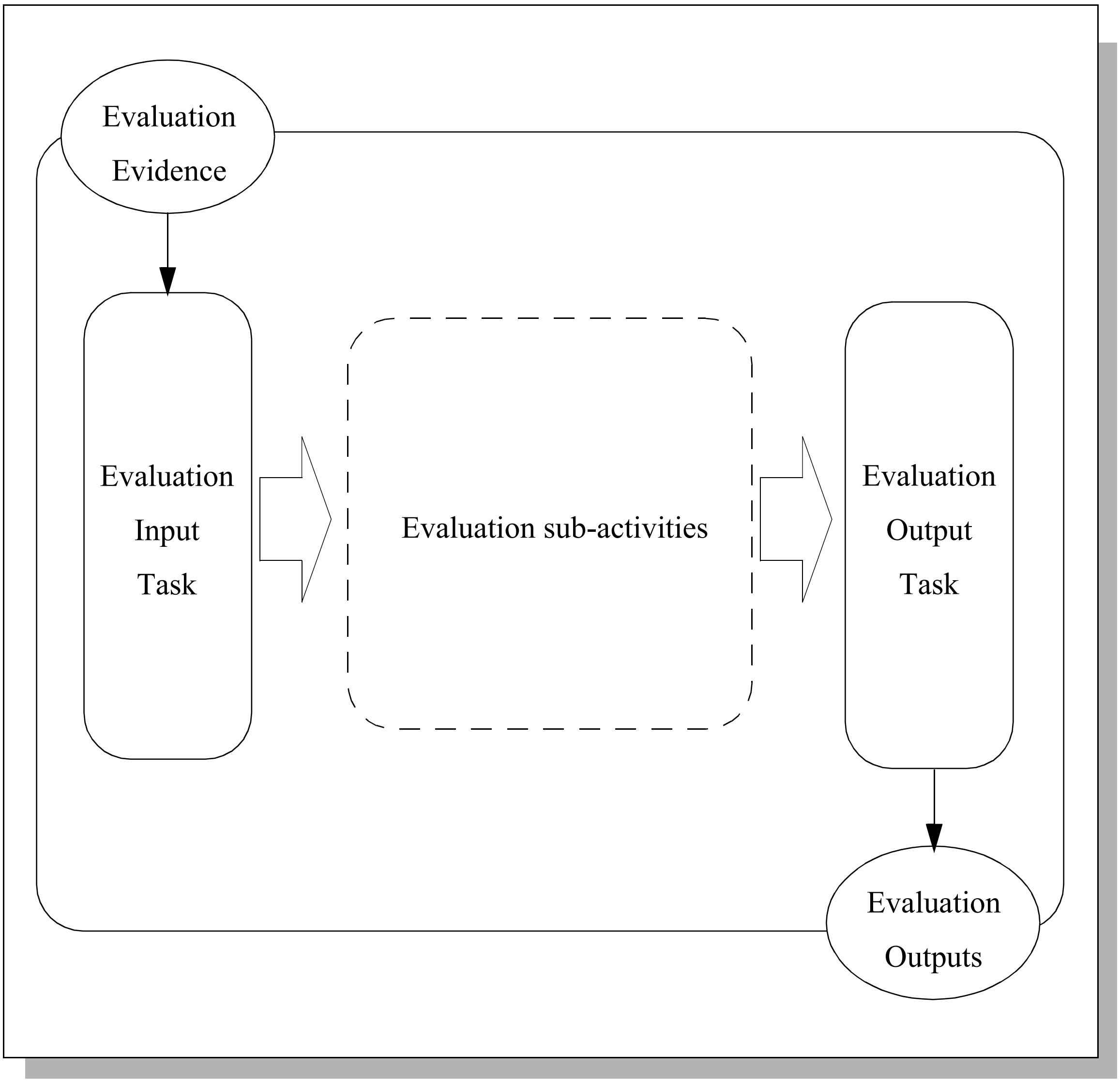

Figure 2.3 Generic evaluation model

Chapter 3

PP evaluation

3.1 Introduction

120 This chapter describes the evaluation of a PP. The requirements and methodology for PP evaluation are identical for each PP evaluation, regardless of the EAL (or other set of assurance criteria) that is claimed in the PP. While further chapters in the CEM are targeted at performing evaluations at specific EALs, this chapter is applicable to any PP that is evaluated.

121 The evaluation methodology in this chapter is based on the requirements of the PP as specified in CC Part 1 especially Annex B, and CC Part 3 class APE.

3.2 Objectives

122 The PP is the description of a product or a system type. As such it is expected to identify the IT security requirements that enforce the defined organisational security policies and counter the defined threats under the defined assumptions.

123 The objective of the PP evaluation is to determine whether the PP is:

a) complete: each threat is countered and each organisational security policy is enforced by the security requirements;

b) sufficient: the IT security requirements are appropriate for the threats and organisational security policies;

c) sound: the PP must be internally consistent.

3.3 PP evaluation relationships

124 The activities to conduct a complete PP evaluation cover the following:

a) evaluation input task (Chapter 2);

b) PP evaluation activity, comprising the following sub-activities:

    1) evaluation of the TOE description (Section 3.4.1);

    2) evaluation of the security environment (Section 3.4.2);

    3) evaluation of the PP introduction (Section 3.4.3);

4) evaluation of the security objectives (Section 3.4.4);

5) evaluation of the IT security requirements (Section 3.4.5);

6) evaluation of the explicitly stated IT security requirements (Section 3.4.6);

c) evaluation output task (Chapter 2).

125 The evaluation input and evaluation output tasks are described in Chapter 2. The evaluation activities are derived from the APE assurance requirements contained in CC Part 3.

126 The sub-activities comprising a PP evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator. For guidance on dependencies see Annex B.4.

127 The evaluation of the explicitly stated IT security requirements sub-activity applies only if security requirements not taken from CC Part 2 or CC Part 3 are included in the IT security requirements statement.

3.4 PP evaluation activity

3.4.1 Evaluation of TOE description (APE_DES.1)

3.4.1.1 Objectives

128 The objective of this sub-activity is to determine whether the TOE description contains relevant information to aid the understanding of the purpose of the TOE and its functionality, and to determine whether the description is complete and consistent.

3.4.1.2 Input

129 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.1.3 Evaluator actions

130 This sub-activity comprises three CC Part 3 evaluator action elements:

a) APE_DES.1.1E;

b) APE_DES.1.2E;

c) APE_DES.1.3E.

3.4.1.3.1 Action APE_DES.1.1E

APE_DES.1.1C

APE_DES.1-1 The evaluator ***shall examine*** the TOE description to determine that it describes the product or system type of the TOE.

131 The evaluator determines that the TOE description is sufficient to give the reader a general understanding of the intended usage of the product or system, thus providing a context for the evaluation. Some examples of product or system types are: firewall, smartcard, crypto-modem, web server, intranet.

132 There are situations where it is clear that some functionality is expected of the TOE because of its product or system type. If this functionality is absent, the evaluator determines whether the TOE description adequately discusses this absence. An example of this is a firewall-type TOE, whose TOE description states that it cannot be connected to networks.

APE_DES.1-2 The evaluator ***shall examine*** the TOE description to determine that it describes the IT features of the TOE in general terms.

133 The evaluator determines that the TOE description discusses the IT, and in particular the security features offered by the TOE at a level of detail that is sufficient to give the reader a general understanding of those features.

3.4.1.3.2 Action APE_DES.1.2E

APE_DES.1-3 The evaluator ***shall examine*** the PP to determine that the TOE description is coherent.

134 The statement of the TOE description is coherent if the text and structure of the statement are understandable by its target audience (i.e. developers, evaluators, and consumers).

APE_DES.1-4 The evaluator ***shall examine*** the PP to determine that the TOE description is internally consistent.

135 The evaluator is reminded that this section of the PP is only intended to define the general intent of the TOE.

136 For guidance on consistency analysis see Annex B.3.

3.4.1.3.3 Action APE_DES.1.3E

APE_DES.1-5 The evaluator ***shall examine*** the PP to determine that the TOE description is consistent with the other parts of the PP.

137 The evaluator determines in particular that the TOE description does not describe threats, security features or configurations of the TOE that are not considered elsewhere in the PP.

138 For guidance on consistency analysis see Annex B.3.

3.4.2 Evaluation of security environment (APE_ENV.1)

3.4.2.1 Objectives

139 The objective of this sub-activity is to determine whether the statement of TOE security environment in the PP provides a clear and consistent definition of the security problem that the TOE and its environment is intended to address.

3.4.2.2 Input

140 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.2.3 Evaluator actions

141 This sub-activity comprises two CC Part 3 evaluator action elements:

a) APE_ENV.1.1E;

b) APE_ENV.1.2E.

3.4.2.3.1 Action APE_ENV.1.1E

APE_ENV.1.1C

APE_ENV.1-1 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any assumptions.

142 The assumptions can be partitioned into assumptions about the intended usage of the TOE, and assumptions about the environment of use of the TOE.

143 The evaluator determines that the assumptions about the intended usage of the TOE address aspects such as the intended application of the TOE, the potential value of the assets requiring protection by the TOE, and possible limitations of use of the TOE.

144 The evaluator determines that each assumption about the intended usage of the TOE is explained in sufficient detail to enable consumers to determine that their intended usage matches the assumption. If the assumptions are not clearly understood, the end result may be that consumers will use the TOE in an environment for which it is not intended.

145 The evaluator determines that the assumptions about the environment of use of the TOE cover the physical, personnel, and connectivity aspects of the environment:

a) Physical aspects include any assumptions that need to be made about the physical location of the TOE or attached peripheral devices in order for the TOE to function in a secure way. Some examples:

- it is assumed that administrator consoles are in an area restricted to only administrator personnel;
- it is assumed that all file storage for the TOE is done on the workstation that the TOE runs on.

b) Personnel aspects include any assumptions that need to be made about users and administrators of the TOE, or other individuals (including potential threat agents) within the environment of the TOE in order for the TOE to function in a secure way. Some examples:

- it is assumed that users have particular skills or expertise;
- it is assumed that users have a certain minimum clearance;
- it is assumed that administrators will update the anti-virus database monthly.

c) Connectivity aspects include any assumptions that need to be made regarding connections between the TOE and other IT systems or products (hardware, software, firmware or a combination thereof) that are external to the TOE in order for the TOE to function in a secure way. Some examples:

- it is assumed that at least 100MB of external disk space is available to store logging files generated by a TOE;
- the TOE is assumed to be the only non-operating system application being executed at a particular workstation;
- the floppy drive of the TOE is assumed to be disabled;
- it is assumed that the TOE will not be connected to an untrusted network.

146 The evaluator determines that each assumption about the environment of use of the TOE is explained in sufficient detail to enable consumers to determine that their intended environment matches the environmental assumption. If the assumptions are not clearly understood, the end result may be that the TOE is used in an environment in which it will not function in a secure manner.

APE_ENV.1.2C

APE_ENV.1-2 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any threats.

147 If the security objectives for the TOE and its environment are derived from assumptions and organisational security policies only, the statement of threats need not be present in the PP. In this case, this work unit is not applicable and therefore considered to be satisfied.

148 The evaluator determines that all identified threats are clearly explained in terms of an identified threat agent, the attack, and the asset that is the subject of the attack.

149 The evaluator also determines that threat agents are characterised by addressing expertise, resources, and motivation and that attacks are characterised by attack methods, any vulnerabilities exploited, and opportunity.

APE_ENV.1.3C

APE_ENV.1-3 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any organisational security policies.

150 If the security objectives for the TOE and its environment are derived from assumptions and threats only, organisational security policies need not be present in the PP. In this case, this work unit is not applicable and therefore considered to be satisfied.

151 The evaluator determines that organisational security policy statements are made in terms of rules, practices or guidelines that must be followed by the TOE or its environment, as laid down by the organisation controlling the environment in which the TOE is to be used. An example organisational security policy is a requirement for password generation and encryption to conform to a standard stipulated by a national government.

152 The evaluator determines that each organisational security policy is explained and/or interpreted in sufficient detail to make it clearly understandable; a clear presentation of policy statements is necessary to permit tracing security objectives to them.

3.4.2.3.2 Action APE_ENV.1.2E

APE_ENV.1-4 The evaluator ***shall examine*** the statement of TOE security environment to determine that it is coherent.

153 The statement of the TOE security environment is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

APE_ENV.1-5 The evaluator ***shall examine*** the statement of TOE security environment to determine that it is internally consistent.

154 Examples of internally inconsistent statements of TOE security environment are:

- a statement of TOE security environment that contains a threat where the attack method is not within the capability of its threat agent;
- a statement of TOE security environment that contains an organisational security policy “The TOE shall not be connected to the Internet” and a threat where the threat agent is an intruder from the Internet.

155 For guidance on consistency analysis see Annex B.3.

3.4.3 Evaluation of PP introduction (APE_INT.1)

3.4.3.1 Objectives

156 The objective of this sub-activity is to determine whether the PP introduction is complete and consistent with all parts of the PP and whether it correctly identifies the PP.

3.4.3.2 Input

157 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.3.3 Evaluator actions

158 This sub-activity comprises three CC Part 3 evaluator action elements:

a) APE_INT.1.1E;

b) APE_INT.1.2E;

c) APE_INT.1.3E.

3.4.3.3.1 Action APE_INT.1.1E

APE_INT.1.1C

APE_INT.1-1 The evaluator ***shall check*** that the PP introduction provides PP identification information necessary to identify, catalogue, register and cross reference the PP.

159 The evaluator determines that the PP identification information includes:

a) information necessary to control and uniquely identify the PP (e.g. title of the PP, version number, publication date, authors, sponsoring organisation);

b) indication of the version of the CC used to develop the PP;

c) registration information, if the PP has been registered before evaluation;

d) cross references, if the PP is compared to other PP(s);

e) additional information, as required by the scheme.

APE_INT.1.2C

APE_INT.1-2 The evaluator ***shall check*** that the PP introduction provides a PP overview in narrative form.

160 The PP overview is intended to provide a brief summary of the content of the PP (a more detailed description is provided in the TOE description) that is sufficiently detailed to enable a potential user of the PP to determine whether the PP is of interest.

3.4.3.3.2 Action APE_INT.1.2E

APE_INT.1-3 The evaluator ***shall examine*** the PP introduction to determine that it is coherent.

161 The PP introduction is coherent if the text and structure of the statement are understandable by its target audience (i.e. developers, evaluators and consumers).

APE_INT.1-4 The evaluator ***shall examine*** the PP introduction to determine that it is internally consistent.

162 The internal consistency analysis will naturally focus on the PP overview that provides a summary of the content of the PP.

163 For guidance on consistency analysis see Annex B.3.

3.4.3.3.3 Action APE_INT.1.3E

APE_INT.1-5 The evaluator ***shall examine*** the PP to determine that the PP introduction is consistent with the other parts of the PP.

164 The evaluator determines that the PP overview provides an accurate summary of the TOE. In particular, the evaluator determines that the PP overview is consistent with the TOE description, and that it does not state or imply the presence of security features that are not in the scope of evaluation.

165 The evaluator also determines that the CC conformance claim is consistent with the rest of the PP.

166 For guidance on consistency analysis see Annex B.3.

3.4.4 Evaluation of security objectives (APE_OBJ.1)

3.4.4.1 Objectives

167 The objective of this sub-activity is to determine whether the security objectives are described completely and consistently, and to determine whether the security objectives counter the identified threats, achieve the identified organisational security policies and are consistent with the stated assumptions.

3.4.4.2 Input

168 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.4.3 Evaluator actions

169 This sub-activity comprises two CC Part 3 evaluator action elements:

a) APE_OBJ.1.1E;

b) APE_OBJ.1.2E.

3.4.4.3.1 Action APE_OBJ.1.1E

APE_OBJ.1.1C

APE_OBJ.1-1 The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the TOE and its environment.

170 The evaluator determines that for each security objective it is clearly specified whether it is intended to apply to the TOE, to the environment, or both.

APE_OBJ.1.2C

APE_OBJ.1-2 The evaluator ***shall examine*** the security objectives rationale to determine that all security objectives for the TOE are traced back to aspects of the identified threats to be countered and/or aspects of the organisational security policies to be met by the TOE.

171 The evaluator determines that each security objective for the TOE is traced back to at least one threat or organisational security policy.

172 Failure to trace implies that either the security objectives rationale is incomplete, the threats or organisational security policy statements are incomplete, or the security objective for the TOE has no useful purpose.

Interp Note : The following paragraph is added as a result of Interpretation 049.

A threat may therefore be addressed entirely by one or more objectives for the environment. An extreme case would be where there are no security objectives for the TOE. Whilst this remains a valid use of the PP/ST construct, a TOE for which all threats and OSPs are addressed by the environment would be of questionable utility, as for such a TOE there would be no security functional requirements for the TOE. Certification/validation of such a TOE is a scheme issue.

APE_OBJ.1.3C

APE_OBJ.1-3 The evaluator ***shall examine*** the security objectives rationale to determine that the security objectives for the environment are traced back to aspects of the identified threats to be countered by the TOE's environment and/or aspects of the organisational security policies to be met by the TOE's environment and/or assumptions to be met in the TOE's environment.

173 The evaluator determines that each security objective for the environment is traced back to at least one assumption, threat or organisational security policy.

174 Failure to trace implies that either the security objectives rationale is incomplete, the threats, assumptions or organisational security policy statements are incomplete, or the security objective for the environment has no useful purpose.

APE_OBJ.1.4C

APE_OBJ.1-4 The evaluator ***shall examine*** the security objectives rationale to determine that for each threat it contains an appropriate justification that the security objectives are suitable to counter that threat.

175 If no security objectives trace back to the threat, this work unit fails.

176 The evaluator determines that the justification for a threat demonstrates that if all security objectives that trace back to the threat are achieved, the threat is removed, the threat is diminished to an acceptable level, or the effects of the threat are sufficiently mitigated.

177 The evaluator also determines that each security objective that traces back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat.

178 Examples of removing a threat are:

- removing the ability to use an attack method from an agent;
- removing the motivation of a threat agent by deterrence;
- removing the threat agent (e.g. removing machines from a network that frequently crash that network).

179 Examples of diminishing a threat are:

- restricting the threat agent in attack methods;
- restricting the threat agents in opportunity;
- reducing the likelihood of a launched attack being successful;
- requiring greater expertise or greater resources from the threat agent.

180 Examples of mitigating the effects of a threat are:

- making frequent back-ups of the asset;
- having spare copies of a TOE;
- frequent changing of keys used in a communication session, so that the effects of breaking one key are relatively minor.

181 Note that the tracings from security objectives to threats provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective is merely a statement reflecting the intent to prevent a particular threat from being realised, a justification is required, but this justification could be quite minimal in this case.

APE_OBJ.1.5C

APE_OBJ.1-5 The evaluator ***shall examine*** the security objectives rationale to determine that for each organisational security policy it contains an appropriate justification that the security objectives are suitable to cover that organisational security policy.

182 If no security objectives trace back to the organisational security policy, this work unit fails.

183 The evaluator determines that the justification for an organisational security policy demonstrates that if all security objectives that trace back to that organisational security policy are achieved, the organisational security policy is implemented.

184 The evaluator also determines that each security objective that traces back to an organisational security policy, when achieved, actually contributes to the implementation of the organisational security policy.

185 Note that the tracings from security objectives to organisational security policies provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective is merely a statement reflecting the intent to implement a particular organisational security policy, a justification is required, but this justification could be quite minimal in this case.

APE_OBJ.1-6 The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption it contains an appropriate justification that the security objectives for the environment are suitable to cover that assumption.

186 If no security objectives for the environment trace back to the assumption, this work unit fails.

187 An assumption is either an assumption about the intended usage of the TOE, or an assumption about the environment of use of the TOE.

188 The evaluator determines that the justification for an assumption about the intended usage of the TOE demonstrates that if all security objectives for the environment that trace back to that assumption are achieved, the intended usage is supported.

189 The evaluator also determines that each security objective for the environment that traces back to an assumption about the intended usage of the TOE, when achieved, actually contributes to the support of the intended usage.

190 The evaluator determines that the justification for an assumption about the environment of use of the TOE demonstrates that if all security objectives for the environment that trace back to that assumption are achieved, the environment is consistent with the assumption.

191 The evaluator also determines that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when

achieved, actually contributes to the environment achieving consistency with the assumption.

192 Note that the tracings from security objectives for the environment to assumptions provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective of the environment is merely a restatement of an assumption, a justification is required, but this justification could be quite minimal in this case.

3.4.4.3.2 Action APE_OBJ.1.2E

APE_OBJ.1-7 The evaluator ***shall examine*** the statement of security objectives to determine that it is coherent.

193 The statement of security objectives is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

APE_OBJ.1-8 The evaluator ***shall examine*** the statement of security objectives to determine that it is complete.

194 The statement of security objectives is complete if the security objectives are sufficient to counter all identified threats, and cover all identified organisational security policies and assumptions. This work unit may be performed in conjunction with the APE_OBJ.1-4, APE_OBJ.1-5 and APE_OBJ.1-6 work units.

APE_OBJ.1-9 The evaluator ***shall examine*** the statement of security objectives to determine that it is internally consistent.

195 The statement of security objectives is internally consistent if the security objectives do not contradict each other. An example of such a contradiction could be two security objectives as “a user’s identity shall never be released”, and “a user’s identity shall be available to the other users”.

196 For guidance on consistency analysis see Annex B.3.

3.4.5 Evaluation of IT security requirements (APE_REQ.1)

3.4.5.1 Objectives

197 The objective of this sub-activity is to determine whether the TOE security requirements (both the TOE security functional requirements and the TOE security assurance requirements) and the security requirements for the IT environment are described completely and consistently, and that they provide an adequate basis for development of a TOE that will achieve its security objectives.

198 Input

199 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.5.2 Evaluator actions

200 This sub-activity comprises two CC Part 3 evaluator action elements:

a) APE_REQ.1.1E;

b) APE_REQ.1.2E.

3.4.5.2.1 Action APE_REQ.1.1E

APE_REQ.1.1C

APE_REQ.1-1 The evaluator **shall check** the statement of TOE security functional requirements to determine that it identifies the TOE security functional requirements drawn from CC Part 2 functional requirements components.

201 The evaluator determines that all TOE security functional requirements components drawn from Part 2 are identified, either by reference to an individual component in Part 2, or by reproduction in the PP.

APE_REQ.1-2 The evaluator ***shall check*** that each reference to a TOE security functional requirement component is correct.

202 The evaluator determines for each reference to a CC Part 2 TOE security functional requirement component whether the referenced component exists in CC Part 2.

APE_REQ.1-3 The evaluator ***shall check*** that each TOE security functional requirement component that was drawn from Part 2 that was reproduced in the PP, is correctly reproduced.

203 The evaluator determines that the requirements are correctly reproduced in the statement of TOE security functional requirements without examination for permitted operations. The examination for correctness of component operations will be performed in the APE_REQ.1-11 work unit.

APE_REQ.1.2C

APE_REQ.1-4 The evaluator **shall check** the statement of TOE security assurance requirements to determine that it identifies the TOE security assurance requirements drawn from CC Part 3 assurance requirements components.

204 The evaluator determines that all TOE security assurance requirements components drawn from Part 3 are identified, either by reference to an EAL, or by reference to an individual component in Part 3, or by reproduction in the PP.

APE_REQ.1-5 The evaluator ***shall check*** that each reference to a TOE security assurance requirement component is correct.

205 The evaluator determines for each reference to a CC Part 3 TOE security assurance requirement component whether the referenced component exists in CC Part 3.

APE_REQ.1-6 The evaluator ***shall check*** that each TOE security assurance requirement component that was drawn from Part 3 that was reproduced in the PP, is correctly reproduced.

206 The evaluator determines that the requirements are correctly reproduced in the statement of TOE security assurance requirements without examination for permitted operations. The examination for correctness of component operations will be performed in the APE_REQ.1-11 work unit.

APE_REQ.1.3C

APE_REQ.1-7 The evaluator ***shall examine*** the statement of TOE security assurance requirements to determine that either it includes an EAL as defined in CC Part 3 or appropriately justifies that it does not include an EAL.

207 If no EAL is included, the evaluator determines that the justification addresses why the statement of TOE assurance requirements contains no EAL. This justification may address the reason why it was impossible, undesirable or inappropriate to include an EAL, or it may address why it was impossible, undesirable or inappropriate to include particular components of the families that constitute EAL1 (ACM_CAP, ADO_IGS, ADV_FSP, ADV_RCR, AGD_ADM, AGD_USR, and ATE_IND).

APE_REQ.1.4C

APE_REQ.1-8 The evaluator ***shall examine*** the security requirements rationale to determine that it sufficiently justifies that the statement of TOE security assurance requirements is appropriate.

208 If the assurance requirements contain an EAL, the justification is allowed to address the choice of that EAL as a whole, rather than addressing all individual components of that EAL. If the assurance requirements contain augmented components to that EAL, the evaluator determines that each augmentation is individually justified. If the assurance requirements contain explicitly stated assurance requirements, the evaluator determines that the use of each explicitly stated assurance requirement is individually justified.

209 The evaluator determines that the security requirements rationale sufficiently justifies that the assurance requirements are sufficient given the statement of security environment and security objectives. For example, if defence against knowledgeable attackers is required, then it would be inappropriate to specify AVA_VLA.1 which is unlikely to detect other than obvious security weaknesses.

210 The justification may also include reasons such as:

a) specific requirements imposed by the scheme, national government, or other organisations;

b) assurance requirements that were dependencies from TOE security functional requirement;

c) assurance requirements of systems and/or products that are to be used in conjunction with a TOE;

d) consumer requirements.

211 An overview of the intent and goals of each EAL is provided in CC Part 3 section 6.2.

212 The evaluator is reminded that determining whether the assurance requirements are appropriate may be subjective and that the analysis of sufficiency of the justification should therefore not be overly rigorous.

213 If the assurance requirements do not contain an EAL, this work unit may be performed in conjunction with the APE_REQ.1-7 work unit.

APE_REQ.1.5C

APE_REQ.1-9 The evaluator ***shall check*** that security requirements for the IT environment are identified, if appropriate.

214 If the PP does not contain security requirements for the IT environment, this work unit is not applicable and therefore considered to be satisfied.

215 The evaluator determines that any dependencies of the TOE on other IT in its environment to provide any security functionality in order for the TOE to achieve its security objectives are clearly identified in the PP as security requirements for the IT environment.

216 An example of a security requirement for the IT environment is a firewall that relies on an underlying operating system to provide authentication of administrators and permanent storage of audit data. In this case, the security requirements for the IT environment would contain components from the FAU and FIA classes.

217 Note that the security requirements for the IT environment can contain both functional and assurance requirements.

218 An example of a dependency on the IT environment is a software crypto-module, which periodically inspects its own code, and disables itself when the code has been tampered with. To allow for recovery, it has the requirement FPT_RCV.2 (automated recovery). As it cannot recover itself once it has disabled itself, this becomes a requirement on the IT environment. One of the dependencies of FPT_RCV.2 is AGD_ADM.1 (administrator guidance). This assurance requirement therefore becomes an assurance requirement for the IT environment.

219 The evaluator is reminded that where security requirements for the IT environment refer to the TSF, they refer to the security functions of the environment, rather than security functions of the TOE.

APE_REQ.1.6C

APE_REQ.1-10 The evaluator ***shall check*** that all completed operations on IT security requirements are identified.

Interp Note : Paragraph 220 is changed as a result of Interpretations 019 and 138.

220 It is permissible for a PP to contain elements with uncompleted operations. That is, the PP can contain IT security ~~functional~~ requirement statements that include uncompleted operations for assignment or selection. The operations have then to be completed in an ST instantiating the PP. This gives the ST developer more flexibility in developing the TOE and the corresponding ST that claims compliance to a particular PP.

Interp Note : Paragraphs 221-222 are replaced as a result of Interpretations 019 and 138.

221 ~~The permitted operations for CC Part 2 functional components are assignment, iteration, selection and refinement. The assignment and selection operations are permitted only where specifically indicated in a component. Iteration and refinement are permitted for all functional components.~~

222 ~~The permitted operations for CC Part 3 assurance components are iteration and refinement.~~

The permitted operations for CC Part 2 and Part 3 components are assignment, iteration, selection and refinement. The assignment and selection operations are permitted only where specifically indicated in a component. Iteration and refinement are permitted for all components.

223 The evaluator determines that all operations are identified in each component where such an operation is used. Completed and uncompleted operations need to be identified in such a way, that they can be distinguished, and that it is clear whether the operation is completed or not. Identification can be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

APE_REQ.1-11 The evaluator ***shall examine*** the statement of IT security requirements to determine that operations are performed correctly.

224 The evaluator is reminded that operations on security requirements need not be performed and completed in a PP.

225 The evaluator compares each statement with the element from which it is derived to determine that:

a) for an assignment, the values of the parameters or variables chosen comply with the indicated type required by the assignment;

b) for a selection, the selected item or items are one or more of the items indicated within the selection portion of the element. The evaluator also determines that the number of items chosen is appropriate for the requirement. Some requirements require a selection of just one item (e.g. FAU_GEN.1.1.b), in other cases multiple items (e.g. FDP_ITT.1.1 second operation) are acceptable.

c) for a refinement, the component is refined in such manner that a TOE meeting the refined requirement also meets the unrefined requirement. If the refined requirement exceeds this boundary it is considered to be an extended requirement.

   Example: ADV_SPM.1.2C The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modelled.
   Refinement: The TSP model need cover only access control.
   If the access control policy is the only policy of the TSP this is a valid refinement. If there are also identification and authentication policies in the TSP, and the refinement is meant to state that only access control needs to be modeled, then this is not a valid refinement.

   A special case of refinement is an editorial refinement, where a small change is made in a requirement, i.e. rephrasing a sentence due to adherence to proper English grammar. This change is not allowed to modify the meaning of the requirement in any way.

   An example of an editorial refinement is FAU_ARP.1 with a single action. Instead of writing: “The TSF shall take *inform the operator* upon detection of a potential security violation” the PP author is allowed to write: “The TSF shall *inform the operator* upon detection of a potential security violation”.

   The evaluator is reminded that editorial refinements have to be clearly identified (see work unit APE_REQ.1-10).

d) for an iteration, that each iteration of a component is different from each other iteration of that component (at least one element of a component is different from the corresponding element of the other component), or that the component applies to a different part of the TOE.

APE_REQ.1.7C

Interp Note : Work unit APE_REQ.1-12 is changed as a result of Interpretation 080.

APE_REQ.1-12 The evaluator ***shall examine*** the statement of IT security requirements to determine that all uncompleted operations ~~on IT security requirements included in the PP~~ are identified.

226 The evaluator determines that all operations are identified in each component where such an operation is used. Completed and uncompleted operations need to be identified in such a way, that they can be distinguished, and that it is clear whether the operation is completed or not. Identification can be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

APE_REQ.1.8C

APE_REQ.1-13 The evaluator ***shall examine*** the statement of IT security requirements to determine that dependencies required by the components used in the IT security requirements statement are satisfied.

227 Dependencies may be satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of TOE security requirements, or as a requirement that is asserted as being met by the IT environment of the TOE.

228 Although the CC provides support for dependency analysis by inclusion of dependency, this is not a justification that no other dependencies exist. An example of such other dependencies is an element that refers to “all objects” or “all subjects”, where a dependency could exist to a refinement in another element or set of elements where the objects or subjects are enumerated.

229 Dependencies of security requirements necessary in the IT environment should be stated and satisfied in the PP.

230 The evaluator is reminded that the CC does not require all dependencies to be satisfied: see the following work-unit.

APE_REQ.1.9C

APE_REQ.1-14 The evaluator ***shall examine*** the security requirements rationale to determine that an appropriate justification is given for each case where security requirement dependencies are not satisfied.

231 The evaluator determines that the justification explains why the dependency is unnecessary, given the identified security objectives.

232 The evaluator confirms that any non-satisfaction of a dependency does not prevent the set of security requirements adequately addressing the security objectives. This analysis is addressed by APE_REQ.1.13C.

233 An example of an appropriate justification is when a software TOE has the security objective: “failed authentications shall be logged with user identity, time and date” and uses FAU_GEN.1 (audit data generation) as a functional requirement to satisfy this security objective. FAU_GEN.1 contains a dependency on FPT_STM.1 (reliable time stamps). As the TOE does not contain a clock mechanism, FPT_STM.1 is defined by the PP author as a requirement on the IT environment. The PP author indicates that this requirement will not be satisfied with the justification: “there are attacks possible on the time-stamping mechanism

in this particular environment, the environment can therefore not deliver a reliable time-stamp. Yet, some threat agents are incapable of executing attacks against the time-stamping mechanisms, and some attacks by these threat agents may be analysed by logging time and date of their attacks."

APE_REQ.1.10C

APE_REQ.1-15 The evaluator ***shall check*** that the PP includes a statement of the minimum strength of function level for the TOE security functional requirements, and that this level is either SOF-basic, SOF-medium or SOF-high.

234 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

235 The strength of cryptographic algorithms is outside the scope of the CC. Strength of function only applies to probabilistic or permutational mechanisms that are non-cryptographic. Therefore, where an PP contains a minimum SOF claim this claim does not apply to any cryptographic mechanisms with respect to a CC evaluation. Where such cryptographic mechanisms are included in a TOE the evaluator determines that the PP includes a clear statement that the assessment of algorithmic strength does not form part of the evaluation.

236 The TOE may contain multiple distinct domains, where the PP writer deems it to be more applicable to have a minimum strength of function level for each domain, rather than having one overall minimum strength of function level for the entire TOE. In this case it is allowed to partition the TOE security functional requirements in distinct sets, and have different minimum strength of function levels associated with each set.

237 An example of this is a distributed terminal system which has user terminals that are in a public space, and administrator terminals that are in a physically secure place. The authentication requirements for the user terminals have SOF-medium associated with them, and the authentication requirements for the administrative terminals have SOF-basic associated with them. Rather than stating that the TOE has a minimum strength of function level of SOF-basic, which might lead potential consumers of the TOE to believe that it would be relatively easy to successfully attack the authentication mechanisms on user terminals, the PP writer divides the TOE into a user domain and an administrative domain, partitions the TOE security functional requirements into sets belonging to those domains, assigns a minimum strength of function level of SOF-basic to the set belonging to the administrative domain, and assigns a minimum strength of function level of SOF-medium to the set belonging to the user domain.

APE_REQ.1.11C

Interp Note : Work unit APE_REQ.1-16 is changed as a result of Interpretation 085.

APE_REQ.1-16 The evaluator ***shall check*** that the PP identifies any specific TOE security functional requirements for which an explicit strength of function is appropriate, together with the specific strength of function or metric as applicable.

238 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

Interp Note : Paragraph 239 is changed as a result of Interpretation 085.

239 The explicit strength of function claim can be either SOF-basic, SOF-medium, SOF-high, or a defined specific metric. Where a specific metric is used, the evaluator determines that these are appropriate for the type of functional requirement specified, and that the metric specified is evaluatable as a strength claim. This work unit refers to the case where a PP author requires to set specific SOF requirements (i.e. higher than the overall SOF claim of the PP) or by using a metric. A specific SOF claim for a TOE security functional requirement may be specified by a PP author. In the absence of any specific claim, the overall claim for the PP applies for all TOE security functional requirements stated in the PP. The evaluator should confirm the presence or absence of explicit SOF claims is consistent with other parts of the PP.

Interp Note : The following paragraph is added as a result of Interpretation 085.

A PP could potentially have varying specifications of SOF claims. There can be an overall SOF claim for a PP and within a PP the TOE security functional requirements could have a SOF claim specified for it.

240 Further guidance on appropriateness and suitability of strength of function metrics may be provided by the scheme.

APE_REQ.1.12C

APE_REQ.1-17 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the minimum strength of function level, together with any explicit strength of function claim, is consistent with the security objectives for the TOE.

241 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

242 The evaluator determines that the rationale takes into account details about the likely expertise, resources, and motivation of attackers as described in the statement of TOE security environment. For example, a claim of SOF-basic is inappropriate if the TOE is required to provide defence against attackers who possess a high attack potential.

243 The evaluator also determines that the rationale takes into account any specific strength-related properties of security objectives. The evaluator can use the tracings from requirements to objectives to determine that requirements that trace towards objectives with specific strength related properties, if appropriate, have a suitable strength of function claim associated with them.

APE_REQ.1.13C

APE_REQ.1-18 The evaluator ***shall examine*** the security requirements rationale to determine that the TOE security requirements are traced back to the security objectives for the TOE.

244 The evaluator determines that each TOE security functional requirement is traced back to at least one security objective for the TOE.

245 Failure to trace implies that either the security requirements rationale is incomplete, the security objectives are incomplete, or that the TOE security functional requirement has no useful purpose.

246 It is also allowed, but not mandatory, for some or all TOE security assurance requirements to trace back to security objectives for the TOE.

247 An example of a TOE security assurance requirement tracing back to a security objective for the TOE is a PP containing the threat “A user unwittingly discloses information by using a device thinking it to be the TOE” and the security objective for the TOE “The TOE shall be clearly labelled with its version number” to counter that threat. This security objective for the TOE can be achieved by satisfying ACM_CAP.1 and the PP author therefore traces ACM_CAP.1 back to that security objective for the TOE.

APE_REQ.1-19 The evaluator ***shall examine*** the security requirements rationale to determine that the security requirements for the IT environment are traced back to the security objectives for the environment.

248 The evaluator determines that each functional security requirement for the IT environment is traced back to at least one security objective for the environment.

249 Failure to trace implies that either the security requirements rationale is incomplete, the security objectives for the environment are incomplete, or that the functional security requirement for the IT environment has no useful purpose.

250 It is also allowed, but not mandatory, for some or all security assurance requirements for the IT environment to trace back to security objectives for the environment.

Interp Note : Work unit APE_REQ.1-20 is changed as a result of Interpretation 084.

APE_REQ.1-20 The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the TOE it contains an appropriate justification that the TOE security requirements are suitable to meet that security objective for the TOE.

251 If no TOE security requirements trace back to the security objective for the TOE, this work unit fails.

252 The evaluator determines that the justification for a security objective for the TOE demonstrates that if all TOE security requirements that trace back to the objective are satisfied, the security objective for the TOE is achieved.

253 The evaluator also determines that each TOE security requirement that traces back to a security objective for the TOE, when satisfied, actually contributes to achieving the security objective.

254 Note that the tracings from TOE security requirements to security objectives for the TOE provided in the security requirements rationale may be a part of the justification, but do not constitute a justification by themselves.

APE_REQ.1-21 The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the IT environment it contains an appropriate justification that the security requirements for the IT environment are suitable to meet that security objective for the IT environment.

255 If no security requirements for the IT environment trace back to the security objective for the IT environment, this work unit fails.

256 The evaluator determines that the justification for a security objective for the environment demonstrates that if all security requirements for the IT environment that trace back to the security objective for the IT environment are satisfied, the security objective for the IT environment is achieved.

257 The evaluator also determines that each security requirement for the IT environment that traces back to a security objective for the IT environment, when satisfied, actually contributes to achieving the security objective.

258 Note that the tracings from security requirements for the IT environment to security objectives for the IT environment provided in the security requirements rationale may be a part of a justification, but do not constitute a justification by themselves.

APE_REQ.1.14C

APE_REQ.1-22 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the set of IT security requirements is internally consistent.

259 The evaluator determines that on all occasions where different IT security requirements apply to the same types of events, operations, data, tests to be performed etc., and these requirements might conflict, an appropriate justification is provided that this is not the case.

260 For example, if the PP contains requirements for individual accountability of users as well as requirements for user anonymity, it needs to be shown that these requirements do not conflict. This might involve showing that none of the auditable events requiring individual user accountability relate to operations for which user anonymity is required.

261 For guidance on consistency analysis see Annex B.3.

APE_REQ.1-23 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the set of IT security requirements together forms a mutually supportive whole.

262 This work unit builds on the determination performed in work units APE_REQ.1-18 and APE_REQ.1-19, which examine the tracing from IT security requirements to security objectives and work units APE_REQ.1-20 and APE_REQ.1-21 which examine whether the IT security requirements are suitable to meet the security objectives. This work unit requires the evaluator to consider the possibility that a security objective might in fact not be achieved because of lack of support from other IT security requirements.

263 This work unit also builds on the dependency analysis addressed by previous work units, because if functional requirement A has a dependency on functional requirement B, B supports A by definition.

264 The evaluator determines that the security requirements rationale demonstrates that functional requirements support each other where necessary, even when no dependency between these requirements is indicated. This demonstration should address security functional requirements that:

a) prevent bypass of other security functional requirements, such as FPT_RVM.1;

b) prevent tampering with other security functional requirements, such as FPT_SEP;

c) prevent de-activation of other security functional requirements, such as FMT_MOF.1;

d) enable detection of attacks aimed at defeating other security functional requirements, such as components of the FAU class.

265 The evaluator takes the performed operations into account in his analysis to determine whether they affect the mutual support between the requirements.

3.4.5.2.2 Action APE_REQ.1.2E

APE_REQ.1-24 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is coherent.

266 The statement of IT security requirements is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

APE_REQ.1-25 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is complete.

267 This work unit draws on the results from the work units required by APE_REQ.1.1E and APE_SRE.1.1E, and in particular the evaluator's examination of the security requirements rationale.

268 The statement of security requirements is complete if the evaluator judges the security requirements to be sufficient to ensure that all security objectives for the TOE are satisfied.

APE_REQ.1-26 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is internally consistent.

269 This work unit draws on the results from the work units required by APE_REQ.1.1E and APE_SRE.1.1E, and in particular the evaluator's examination of the security requirements rationale.

270 The statement of security requirements is internally consistent if the evaluator determines that no security requirement conflicts with any other security requirement, such that a security objective will not be fully satisfied.

271 For guidance on consistency analysis see Annex B.3.

3.4.6 Evaluation of explicitly stated IT security requirements (APE_SRE.1)

3.4.6.1 Objectives

272 The objective of this sub-activity is to determine whether the security functional requirements or security assurance requirements that are stated without reference to the CC are appropriate and adequate.

3.4.6.2 Application Notes

273 This section is only applicable if the PP contains IT security requirements that are explicitly stated without reference to either CC Part 2 or CC Part 3. If this is not the case, all work units in this section are not applicable, and therefore considered to be satisfied.

274 The APE_SRE requirements do not replace the APE_REQ requirements, but are additional to them. This means that IT security requirements that are explicitly stated without reference to either CC Part 2 or CC Part 3 must be evaluated with the APE_SRE criteria, and also, in combination with all other security requirements, with the APE_REQ criteria.

3.4.6.3 Input

275 The evaluation evidence for this sub-activity is:

a) the PP.

3.4.6.4 Evaluator actions

276 This sub-activity comprises two CC Part 3 evaluator action elements:

a) APE_SRE.1.1E;

b) APE_SRE.1.2E.

3.4.6.4.1 Action APE_SRE.1.1E

APE_SRE.1.1C

APE_SRE.1-1 The evaluator ***shall check*** that the statement of the IT security requirements identifies all TOE security requirements that are explicitly stated without reference to the CC.

277 Any TOE security functional requirements that are not specified using CC Part 2 functional components are required to be clearly identified as such. Similarly, any TOE security assurance requirements that are not specified using CC Part 3 assurance components are also required to be clearly identified as such.

APE_SRE.1.2C

APE_SRE.1-2 The evaluator ***shall check*** that the statement of IT security requirements identifies all security requirements for the IT environment that are explicitly stated without reference to the CC.

278 Any security functional requirements for the IT environment that are not specified using CC Part 2 functional components are required to be clearly identified as such. Similarly, any security assurance requirements for the IT environment that are not specified using CC Part 3 assurance components are also required to be clearly identified as such.

APE_SRE.1.3C

APE_SRE.1-3 The evaluator ***shall examine*** the security requirements rationale to determine that it appropriately justifies why each explicitly stated IT security requirement had to be explicitly stated.

279 The evaluator determines for each explicitly stated IT security requirement that the justification explains why existing functional or assurance components (from CC Part 2 and CC Part 3, respectively) could not be used to express the explicitly stated security requirement in question. The evaluator takes the possibility of performing operations (i.e. assignment, iteration, selection or refinement) on these existing components into account in this determination.

APE_SRE.1.4C

APE_SRE.1-4 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that the requirement uses the CC requirements components, families and classes as a model for presentation.

280 The evaluator determines that explicitly stated IT security requirements are presented in the same style as CC Part 2 or CC Part 3 components and to a comparable level of detail. The evaluator also determines that the functional requirements are broken down into individual functional elements and that the assurance requirements specify the developer action, content and presentation of evidence, and evaluator action elements.

APE_SRE.1.5C

APE_SRE.1-5 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that it is measurable and states objective evaluation requirements, such that compliance or noncompliance of a TOE can be determined and systematically demonstrated.

281 The evaluator determines that functional requirements are stated in such a way that they are testable, and traceable through the appropriate TSF representations. The evaluator also determines that assurance requirements avoid the need for subjective evaluator judgement.

Interp Note : The following paragraph is added as a result of Interpretation 064.

The existing CC functional and assurance requirements are to be used as models for compliance with this requirement.

APE_SRE.1.6C

APE_SRE.1-6 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that it is clearly and unambiguously expressed.

Interp Note : The following paragraph is added as a result of Interpretation 064.

The existing CC functional and assurance requirements are to be used as models for compliance with this requirement.

APE_SRE.1.7C

APE_SRE.1-7 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the assurance requirements are applicable and appropriate to support any explicitly stated TOE security functional requirements.

282 The evaluator determines whether application of the specified assurance requirements will yield a meaningful evaluation result for each explicitly stated security functional requirement, or whether other assurance requirements should have been specified. For example, an explicitly stated functional requirement may imply the need for particular documentary evidence (such as a TSP model), depth

of testing, or analysis (such as strength of TOE security functions analysis or covert channel analysis).

3.4.6.4.2 Action APE_SRE.1.2E

APE_SRE.1-8 The evaluator ***shall examine*** the statement of IT security requirements to determine that all of the dependencies of any explicitly stated IT security requirement have been identified.

283 The evaluator confirms that no applicable dependencies have been overlooked by the PP author.

284 Examples of possible dependencies are: components of the FAU class if an explicitly stated functional requirement mentions auditing and ADV_IMP if an explicitly stated assurance requirement mentions the source code or implementation representation of the TOE.

Chapter 4

ST evaluation

4.1 Introduction

285 This chapter describes the evaluation of an ST. The ST evaluation is started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities. A final verdict on the ST may not be possible until the TOE evaluation is complete, since changes to the ST may result from sub-activity findings in the TOE evaluation.

286 The requirements and methodology for ST evaluation are identical for each ST evaluation, regardless of the EAL (or other set of assurance criteria) that is claimed in the ST. While further chapters in the CEM are targeted at performing evaluations at specific EALs, this chapter is applicable to any ST that is evaluated.

287 The evaluation methodology in this chapter is based on the requirements of the ST as specified in CC Part 1 especially Annex C, and CC Part 3 class ASE.

4.2 Objectives

288 The ST is the description of a product or a system. As such it is expected to identify the security functions, and possibly the security mechanisms that enforce the defined organisational security policies and counter the defined threats under the defined assumptions. It is also expected to define the measures that provide the assurance that the product or system correctly counters the threats and enforces the organisational security policies.

289 The objective of the ST evaluation is to determine whether the ST is:

a) complete: each threat is countered and each organisational security policy is enforced by the security functions;

b) sufficient: the security functions are appropriate for the threats and organisational security policies, and the assurance measures provide sufficient assurance that the security functions are correctly implemented;

c) sound: the ST must be internally consistent;

d) accurately instantiated: if the ST claims to satisfy one or more PPs, then the ST must be a complete and accurate instantiation of each referenced PP. In this case many of the evaluation results of the PP may be re-used in evaluating the ST.

4.3 ST evaluation relationships

290 The activities to conduct a complete ST evaluation cover the following:

a) evaluation input task (Chapter 2);

b) ST evaluation activity, comprising the following sub-activities:

1) evaluation of the TOE description (Section 4.4.1);

2) evaluation of the security environment (Section 4.4.2);

3) evaluation of the ST introduction (Section 4.4.3);

4) evaluation of the security objectives (Section 4.4.4);

5) evaluation of the PP claims (Section 4.4.5);

6) evaluation of the IT security requirements (Section 4.4.6);

7) evaluation of the explicitly stated IT security requirements (Section 4.4.7);

8) evaluation of the TOE summary specification (Section 4.4.8).

c) evaluation output task (Chapter 2).

291 The evaluation input and evaluation output tasks are described in Chapter 2. The evaluation activities are derived from the ASE assurance requirements contained in CC Part 3.

292 The sub-activities comprising an ST evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator. For guidance on dependencies see Annex B.4.

293 The evaluation of the PP claims and the evaluation of the explicitly stated IT security requirements sub-activities do not always have to be performed: the evaluation of the PP claims sub-activity applies only if a PP claim is made, and the evaluation of the explicitly stated IT security requirements sub-activity applies only if security requirements not taken from CC Part 2 or CC Part 3 are included in the IT security requirements statement.

294 Some of the information required for the ST may be included by reference. For example if compliance to a PP is claimed, the information in the PP such as the information about the environment and threats is considered to be part of the ST and should conform to the criteria for the ST.

295 If the ST claims compliance with an evaluated PP, and is largely based on the content of that PP, then it may be possible to reuse the PP evaluation results in

performing many of the sub-activities listed above. In particular, reuse may be possible when evaluating the statement of security environment, the security objectives and IT security requirements. It is allowed for an ST to claim compliance with multiple PPs.

4.4 ST evaluation activity

4.4.1 Evaluation of TOE description (ASE_DES.1)

4.4.1.1 Objectives

296 The objective of this sub-activity is to determine whether the TOE description contains relevant information to aid the understanding of the purpose of the TOE and its functionality, and to determine whether the description is complete and consistent.

4.4.1.2 Input

297 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.1.3 Application notes

298 There may be a difference between a TOE and a product that a consumer might purchase. A discussion on this subject can be found in Annex B.6.

4.4.1.4 Evaluator actions

299 This sub-activity comprises three CC Part 3 evaluator action elements:

a) ASE_DES.1.1E;

b) ASE_DES.1.2E;

c) ASE_DES.1.3E.

4.4.1.4.1 Action ASE_DES.1.1E

ASE_DES.1.1C

ASE_DES.1-1 The evaluator ***shall examine*** the TOE description to determine that it describes the product or system type of the TOE.

300 The evaluator determines that the TOE description is sufficient to give the reader a general understanding of the intended usage of the product or system, thus providing a context for the evaluation. Some examples of product or system types are: firewall, smartcard, crypto-modem, web server, intranet.

301 There are situations where it is clear that some functionality is expected of the TOE because of its product or system type. If this functionality is absent, the evaluator determines whether the TOE description adequately discusses this absence. An example of this is a firewall-type TOE, whose TOE description states that it cannot be connected to networks.

ASE_DES.1-2 The evaluator ***shall examine*** the TOE description to determine that it describes the physical scope and boundaries of the TOE in general terms.

302 The evaluator determines that the TOE description discusses the hardware, firmware and software components and/or modules that constitute the TOE at a level of detail that is sufficient to give the reader a general understanding of those components and/or modules.

303 If the TOE is not identical to a product, the evaluator determines that the TOE description adequately describes the physical relationship between the TOE and the product.

ASE_DES.1-3 The evaluator ***shall examine*** the TOE description to determine that it describes the logical scope and boundaries of the TOE in general terms.

304 The evaluator determines that the TOE description discusses the IT, and in particular the security features offered by the TOE at a level of detail that is sufficient to give the reader a general understanding of those features.

305 If the TOE is not identical to a product, the evaluator determines that the TOE description adequately describes the logical relationship between the TOE and the product.

4.4.1.4.2 Action ASE_DES.1.2E

ASE_DES.1-4 The evaluator ***shall examine*** the ST to determine that the TOE description is coherent.

306 The statement of the TOE description is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

ASE_DES.1-5 The evaluator ***shall examine*** the ST to determine that the TOE description is internally consistent.

307 The evaluator is reminded that this section of the ST is only intended to define the general intent of the TOE.

308 For guidance on consistency analysis see Annex B.3.

4.4.1.4.3 Action ASE_DES.1.3E

ASE_DES.1-6 The evaluator ***shall examine*** the ST to determine that the TOE description is consistent with the other parts of the ST.

309 The evaluator determines in particular that the TOE description does not describe threats, security features or configurations of the TOE that are not considered elsewhere in the ST.

310 For guidance on consistency analysis see Annex B.3.

4.4.2 Evaluation of security environment (ASE_ENV.1)

4.4.2.1 Objectives

311 The objective of this sub-activity is to determine whether the statement of TOE security environment in the ST provides a clear and consistent definition of the security problem that the TOE and its environment is intended to address.

4.4.2.2 Input

312 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.2.3 Evaluator actions

313 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_ENV.1.1E;

b) ASE_ENV.1.2E.

4.4.2.3.1 Action ASE_ENV.1.1E

ASE_ENV.1.1C

ASE_ENV.1-1 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any assumptions.

314 The assumptions can be partitioned into assumptions about the intended usage of the TOE, and assumptions about the environment of use of the TOE.

315 The evaluator determines that the assumptions about the intended usage of the TOE address aspects such as the intended application of the TOE, the potential value of the assets requiring protection by the TOE, and possible limitations of use of the TOE.

316 The evaluator determines that each assumption about the intended usage of the TOE is explained in sufficient detail to enable consumers to determine that their intended usage matches the assumption. If the assumptions are not clearly understood, the end result may be that consumers will use the TOE in an environment for which it is not intended.

317 The evaluator determines that the assumptions about the environment of use of the TOE cover the physical, personnel, and connectivity aspects of the environment:

a) Physical aspects include any assumptions that need to be made about the physical location of the TOE or attached peripheral devices in order for the TOE to function in a secure way. Some examples:

- it is assumed that administrator consoles are in an area restricted to only administrator personnel;
- it is assumed that all file storage for the TOE is done on the workstation that the TOE runs on.

b) Personnel aspects include any assumptions that need to be made about users and administrators of the TOE, or other individuals (including potential threat agents) within the environment of the TOE in order for the TOE to function in a secure way. Some examples:

- it is assumed that users have particular skills or expertise;
- it is assumed that users have a certain minimum clearance;
- it is assumed that administrators will update the anti-virus database monthly.

c) Connectivity aspects include any assumptions that need to be made regarding connections between the TOE and other IT systems or products (hardware, software, firmware or a combination thereof) that are external to the TOE in order for the TOE to function in a secure way. Some examples:

- it is assumed that at least 100MB of external disk space is available to store logging files generated by a TOE;
- the TOE is assumed to be the only non-operating system application being executed at a particular workstation;
- the floppy drive of the TOE is assumed to be disabled;
- it is assumed that the TOE will not be connected to an untrusted network.

318 The evaluator determines that each assumption about the environment of use of the TOE is explained in sufficient detail to enable consumers to determine that their intended environment matches the environmental assumption. If the assumptions are not clearly understood, the end result may be that the TOE is used in an environment in which it will not function in a secure manner.

ASE_ENV.1.2C

ASE_ENV.1-2 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any threats.

319 If the security objectives for the TOE and its environment are derived from assumptions and organisational security policies only, the statement of threats need not be present in the ST. In this case, this work unit is not applicable and therefore considered to be satisfied.

320 The evaluator determines that all identified threats are clearly explained in terms of an identified threat agent, the attack, and the asset that is the subject of the attack.

321 The evaluator also determines that threat agents are characterised by addressing expertise, resources, and motivation and that attacks are characterised by attack methods, any vulnerabilities exploited, and opportunity.

ASE_ENV.1.3C

ASE_ENV.1-3 The evaluator ***shall examine*** the statement of TOE security environment to determine that it identifies and explains any organisational security policies.

322 If the security objectives for the TOE and its environment are derived from assumptions and threats only, organisational security policies need not be present in the ST. In this case, this work unit is not applicable and therefore considered to be satisfied.

323 The evaluator determines that organisational security policy statements are made in terms of rules, practices or guidelines that must be followed by the TOE or its environment, as laid down by the organisation controlling the environment in which the TOE is to be used. An example organisational security policy is a requirement for password generation and encryption to conform to a standard stipulated by a national government.

324 The evaluator determines that each organisational security policy is explained and/or interpreted in sufficient detail to make it clearly understandable; a clear presentation of policy statements is necessary to permit tracing security objectives to them.

4.4.2.3.2 Action ASE_ENV.1.2E

ASE_ENV.1-4 The evaluator ***shall examine*** the statement of TOE security environment to determine that it is coherent.

325 The statement of the TOE security environment is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

ASE_ENV.1-5 The evaluator ***shall examine*** the statement of TOE security environment to determine that it is internally consistent.

326 Examples of internally inconsistent statements of TOE security environment are:

- a statement of TOE security environment that contains a threat where the attack method is not within the capability of its threat agent;
- a statement of TOE security environment that contains an organisational security policy “The TOE shall not be connected to the Internet” and a threat where the threat agent is an intruder from the Internet.

327 For guidance on consistency analysis see Annex B.3.

4.4.3 Evaluation of ST introduction (ASE_INT.1)

4.4.3.1 Objectives

328 The objective of this sub-activity is to determine whether the ST introduction is complete and consistent with all parts of the ST and whether it correctly identifies the ST.

4.4.3.2 Input

329 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.3.3 Evaluator actions

330 This sub-activity comprises three CC Part 3 evaluator action elements:

a) ASE_INT.1.1E;

b) ASE_INT.1.2E;

c) ASE_INT.1.3E.

4.4.3.3.1 Action ASE_INT.1.1E

ASE_INT.1.1C

ASE_INT.1-1 The evaluator ***shall check*** that the ST introduction provides ST identification information necessary to control and identify the ST and the TOE to which it refers.

331 The evaluator determines that the ST identification information includes:

a) information necessary to control and uniquely identify the ST (e.g. title of the ST, version number, publication date, authors);

b) information necessary to control and uniquely identify the TOE to which the ST refers (e.g. identity of the TOE, version number of the TOE);

c) indication of the version of the CC used to develop the ST;

d) additional information, as required by the scheme.

ASE_INT.1.2C

ASE_INT.1-2 The evaluator ***shall check*** that the ST introduction provides an ST overview in narrative form.

332 The ST overview is intended to provide a brief summary of the content of the ST (a more detailed description is provided in the TOE description) that is sufficiently detailed to enable a potential consumer to determine whether the TOE (and therefore the rest of the ST) is of interest.

ASE_INT.1.3C

ASE_INT.1-3 The evaluator ***shall check*** that the ST introduction contains a CC conformance claim that states a claim of CC conformance for the TOE.

333 The evaluator determines that the CC conformance claim is in accordance with section 5.4 of CC Part 1.

334 The evaluator determines that the CC conformance claim contains either Part 2 conformant or Part 2 extended.

Interp Note : Paragraphs 335-340 are replaced as a result of Interpretation 008.

335 ~~The evaluator determines that the CC conformance claim contains either Part 3 conformant or one or both of Part 3 augmented and Part 3 extended.~~

336 ~~If Part 3 conformant is claimed, the evaluator determines that the CC conformance claim states which EAL or assurance package is claimed.~~

337 ~~If Part 3 augmented is claimed, the evaluator determines that the CC conformance claim states which EAL or assurance package is claimed and which augmentations to that EAL or assurance package are claimed.~~

338 ~~If Part 3 extended is claimed and the assurance requirements are in the form of an EAL associated with additional assurance requirements not in Part 3, the evaluator determines that the CC conformance claim states which EAL is claimed.~~

339 ~~If Part 3 extended is claimed and the assurance requirements are in the form of an assurance package that includes assurance requirements not in Part 3, the evaluator determines that the CC conformance claim states which assurance requirements that are in Part 3 are claimed.~~

340 ~~If conformance to a PP is claimed, the evaluator determines that the CC conformance claim states to which PP or PPs conformance is claimed.~~

The evaluator determines that the CC conformance claim contains either Part 3 conformant or Part 3 extended.

If Part 3 extended is claimed and the assurance requirements package includes assurance requirements in Part 3, the evaluator determines that the CC conformance claim states which assurance requirements that are in Part 3 are claimed.

If Package Name conformant is claimed, the evaluator determines that the CC conformance claim states which package is claimed.

If Package Name augmented is claimed, the evaluator determines that the CC conformance claim states which package is claimed and which augmentations to that package are claimed.

If PP conformant is claimed, the evaluator determines that the CC conformance claim states to which PP or PPs conformance is claimed.

341 The evaluator is reminded that if conformance to a PP is claimed the ASE_PPC.1 criteria apply and that if either Part 2 extended or Part 3 extended is claimed the ASE_SRE.1 criteria apply.

4.4.3.3.2 Action ASE_INT.1.2E

ASE_INT.1-4 The evaluator ***shall examine*** the ST introduction to determine that it is coherent.

342 The ST introduction is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

ASE_INT.1-5 The evaluator ***shall examine*** the ST introduction to determine that it is internally consistent.

343 The internal consistency analysis will naturally focus on the ST overview that provides a summary of the content of the ST.

344 For guidance on consistency analysis see Annex B.3.

4.4.3.3.3 Action ASE_INT.1.3E

ASE_INT.1-6 The evaluator ***shall examine*** the ST to determine that the ST introduction is consistent with the other parts of the ST.

345 The evaluator determines that the ST overview provides an accurate summary of the TOE. In particular, the evaluator determines that the ST overview is consistent with the TOE description, and that it does not state or imply the presence of security features that are not in the scope of evaluation.

346 The evaluator also determines that the CC conformance claim is consistent with the rest of the ST.

347 For guidance on consistency analysis see Annex B.3.

4.4.4 Evaluation of security objectives (ASE_OBJ.1)

4.4.4.1 Objectives

348 The objective of this sub-activity is to determine whether the security objectives are described completely and consistently, and to determine whether the security objectives counter the identified threats, achieve the identified organisational security policies and are consistent with the stated assumptions.

4.4.4.2 Input

349 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.4.3 Evaluator actions

350 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_OBJ.1.1E;

b) ASE_OBJ.1.2E.

4.4.4.3.1 Action ASE_OBJ.1.1E

ASE_OBJ.1.1C

ASE_OBJ.1-1 The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the TOE and its environment.

351 The evaluator determines that for each security objective it is clearly specified whether it is intended to apply to the TOE, to the environment, or both.

ASE_OBJ.1.2C

ASE_OBJ.1-2 The evaluator ***shall examine*** the security objectives rationale to determine that all security objectives for the TOE are traced back to aspects of the identified threats to be countered and/or aspects of the organisational security policies to be met by the TOE.

352 The evaluator determines that each security objective for the TOE is traced back to at least one threat or organisational security policy.

353 Failure to trace implies that either the security objectives rationale is incomplete, the threats or organisational security policy statements are incomplete, or the security objective for the TOE has no useful purpose.

ASE_OBJ.1.3C

ASE_OBJ.1-3 The evaluator ***shall examine*** the security objectives rationale to determine that the security objectives for the environment are traced back to aspects of the identified threats to be countered by the TOE's environment and/or aspects of the organisational security policies to be met by the TOE's environment and/or assumptions to be met in the TOE's environment.

354 The evaluator determines that each security objective for the environment is traced back to at least one assumption, threat or organisational security policy.

355 Failure to trace implies that either the security objectives rationale is incomplete, the threats, assumptions or organisational security policy statements are incomplete, or the security objective for the environment has no useful purpose.

Interp Note : The following paragraph is added as a result of Interpretation 049.

A threat may therefore be addressed entirely by one or more objectives for the environment. An extreme case would be where there are no security objectives for the TOE. Whilst this remains a valid use of the PP/ST construct, a TOE for which all threats and OSPs are addressed by the environment would be of questionable utility, as for such a TOE there would be no security functional requirements for the TOE. Certification/validation of such a TOE is a scheme issue.

ASE_OBJ.1.4C

ASE_OBJ.1-4 The evaluator ***shall examine*** the security objectives rationale to determine that for each threat it contains an appropriate justification that the security objectives are suitable to counter that threat.

356 If no security objectives trace back to the threat, this work unit fails.

357 The evaluator determines that the justification for a threat demonstrates that if all security objectives that trace back to the threat are achieved, the threat is removed, the threat is diminished to an acceptable level, or the effects of the threat are sufficiently mitigated.

358 The evaluator also determines that each security objective that traces back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat.

359 Examples of removing a threat are:

- removing the ability to use an attack method from an agent;
- removing the motivation of a threat agent by deterrence;
- removing the threat agent (e.g. removing machines from a network that frequently crash that network).

360 Examples of diminishing a threat are:

- restricting the threat agent in attack methods;
- restricting the threat agents in opportunity;
- reducing the likelihood of a launched attack being successful;
- requiring greater expertise or greater resources from the threat agent.

361 Examples of mitigating the effects of a threat are:

- making frequent back-ups of the asset;
- having spare copies of a TOE;
- frequent changing of keys used in a communication session, so that the effects of breaking one key are relatively minor.

362 Note that the tracings from security objectives to threats provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective is merely a statement reflecting the intent to prevent a particular threat from being realised, a justification is required, but this justification could be quite minimal in this case.

ASE_OBJ.1.5C

ASE_OBJ.1-5 The evaluator ***shall examine*** the security objectives rationale to determine that for each organisational security policy it contains an appropriate justification that the security objectives are suitable to cover that organisational security policy.

363 If no security objectives trace back to the organisational security policy, this work unit fails.

364 The evaluator determines that the justification for an organisational security policy demonstrates that if all security objectives that trace back to that organisational security policy are achieved, the organisational security policy is implemented.

365 The evaluator also determines that each security objective that traces back to an organisational security policy, when achieved, actually contributes to the implementation of the organisational security policy.

366 Note that the tracings from security objectives to organisational security policies provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective is merely a statement reflecting the intent to implement a particular organisational security policy, a justification is required, but this justification could be quite minimal in this case.

ASE_OBJ.1-6 The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption it contains an appropriate justification that the security objectives for the environment are suitable to cover that assumption.

367 If no security objectives for the environment trace back to the assumption, this work unit fails.

368 An assumption is either an assumption about the intended usage of the TOE, or an assumption about the environment of use of the TOE.

369 The evaluator determines that the justification for an assumption about the intended usage of the TOE demonstrates that if all security objectives for the environment that trace back to that assumption are achieved, the intended usage is supported.

370 The evaluator also determines that each security objective for the environment that traces back to an assumption about the intended usage of the TOE, when achieved, actually contributes to the support of the intended usage.

371 The evaluator determines that the justification for an assumption about the environment of use of the TOE demonstrates that if all security objectives for the environment that trace back to that assumption are achieved, the environment is consistent with the assumption.

372 The evaluator also determines that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption.

373 Note that the tracings from security objectives for the environment to assumptions provided in the security objectives rationale may be a part of a justification, but do not constitute a justification by themselves. Even in the case that a security objective of the environment is merely a restatement of an assumption, a justification is required, but this justification could be quite minimal in this case.

4.4.4.3.2 Action ASE_OBJ.1.2E

ASE_OBJ.1-7 The evaluator ***shall examine*** the statement of security objectives to determine that it is coherent.

374 The statement of security objectives is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

ASE_OBJ.1-8 The evaluator ***shall examine*** the statement of security objectives to determine that it is complete.

375 The statement of security objectives is complete if the security objectives are sufficient to counter all identified threats, and cover all identified organisational security policies and assumptions. This work unit may be performed in conjunction with the ASE_OBJ.1-4, ASE_OBJ.1-5 and ASE_OBJ.1-6 work units.

ASE_OBJ.1-9 The evaluator ***shall examine*** the statement of security objectives to determine that it is internally consistent.

376 The statement of security objectives is internally consistent if the security objectives do not contradict each other. An example of such a contradiction could be two security objectives as “a user’s identity shall never be released”, and “a user’s identity shall be available to the other users”.

377 For guidance on consistency analysis see Annex B.3.

4.4.5 Evaluation of PP claims (ASE_PPC.1)

378 This section is only applicable if the ST claims compliance with one or more PPs. If the ST does not claim compliance with one or more PPs, all work units in this section are not applicable, and therefore considered to be satisfied.

4.4.5.1 Objectives

379 The objective of this sub-activity is to determine whether the ST is a correct instantiation of any PP for which compliance is being claimed.

4.4.5.2 Input

380 The evaluation evidence for this sub-activity is:

a) the ST;

b) the PP(s) that the ST claims compliance to.

4.4.5.3 Evaluator actions

381 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_PPC.1.1E;

b) ASE_PPC.1.2E.

4.4.5.3.1 Action ASE_PPC.1.1E

ASE_PPC.1.1C

ASE_PPC.1-1 The evaluator ***shall check*** that each PP claim identifies the PP for which compliance is being claimed.

382 The evaluator determines that any referenced PPs are unambiguously identified (e.g. by title and version number, or by the identification included in the introduction of that PP). The evaluator is reminded that claims of partial compliance to a PP are not permitted under the CC.

ASE_PPC.1.2C

ASE_PPC.1-2 The evaluator ***shall check*** that each PP claim identifies the IT security requirements statements that satisfy the permitted operations of the PP or otherwise further qualify the PP requirements.

383 The ST does not need to repeat statements of security requirements that are included in a PP that are unmodified for this ST. If, however, the PP security functional requirements include uncompleted operations, or the ST author has applied the refinement operation on any PP security requirement, then these requirements in the ST must be clearly identified.

ASE_PPC.1.3C

ASE_PPC.1-3 The evaluator ***shall check*** that each PP claim identifies those security objectives and IT security requirements that are additional to the security objectives and the IT security requirements contained in the PP.

384 The evaluator determines that all security objectives and security requirements that are included in the ST, but were not included in the PP, are clearly identified.

4.4.5.3.2 Action ASE_PPC.1.2E

ASE_PPC.1-4 For each PP claim, the evaluator ***shall examine*** the ST to determine that all operations that were performed on the IT security requirements from the PP are within the bounds set by the PP.

385 This work unit covers not only the uncompleted assignment or selection operations in the PP, but also any application of the refinement operation on the security requirements taken from the PP.

4.4.6 Evaluation of IT security requirements (ASE_REQ.1)

4.4.6.1 Objectives

386 The objective of this sub-activity is to determine whether the TOE security requirements (both the TOE security functional requirements and the TOE security assurance requirements) and the security requirements for the IT environment are described completely and consistently, and that they provide an adequate basis for development of a TOE that will achieve its security objectives.

4.4.6.2 Input

387 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.6.3 Evaluator actions

388 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_REQ.1.1E;

b) ASE_REQ.1.2E.

4.4.6.3.1 Action ASE_REQ.1.1E

ASE_REQ.1.1C

ASE_REQ.1-1 The evaluator **shall check** the statement of TOE security functional requirements to determine that it identifies the TOE security functional requirements drawn from CC Part 2 functional requirements components.

389 The evaluator determines that all TOE security functional requirements components drawn from Part 2 are identified, either by reference to an individual component in Part 2, or by reference to an individual component in a PP that the ST claims to be compliant with, or by reproduction in the ST.

ASE_REQ.1-2 The evaluator ***shall check*** that each reference to a TOE security functional requirement component is correct.

390 The evaluator determines for each reference to a CC Part 2 TOE security functional requirement component whether the referenced component exists in CC Part 2.

391 The evaluator determines for each reference to a TOE security functional requirement component in a PP whether the referenced component exists in that PP.

ASE_REQ.1-3 The evaluator ***shall check*** that each TOE security functional requirement component that was drawn from Part 2 that was reproduced in the ST, is correctly reproduced.

392 The evaluator determines that the requirements are correctly reproduced in the statement of TOE security functional requirements without examination for permitted operations. The examination for correctness of component operations will be performed in the ASE_REQ.1-11 and ASE_REQ.1-12 work units.

ASE_REQ.1.2C

ASE_REQ.1-4 The evaluator **shall check** the statement of TOE security assurance requirements to determine that it identifies the TOE security assurance requirements drawn from CC Part 3 assurance requirements components.

393 The evaluator determines that all TOE security assurance requirements components drawn from Part 3 are identified, either by reference to an EAL, or by reference to an individual component in Part 3, or by reference to a PP that the ST claims to be compliant with, or by reproduction in the ST.

ASE_REQ.1-5 The evaluator ***shall check*** that each reference to a TOE security assurance requirement component is correct.

394 The evaluator determines for each reference to a CC Part 3 TOE security assurance requirement component whether the referenced component exists in CC Part 3.

395 The evaluator determines for each reference to a TOE security assurance requirement component in a PP whether the referenced component exists in that PP.

ASE_REQ.1-6 The evaluator ***shall check*** that each TOE security assurance requirement component that was drawn from Part 3 that was reproduced in the ST, is correctly reproduced.

396 The evaluator determines that the requirements are correctly reproduced in the statement of TOE security assurance requirements without examination for permitted operations. The examination for correctness of component operations will be performed in the ASE_REQ.1-11 and ASE_REQ.1-12 work units.

ASE_REQ.1.3C

ASE_REQ.1-7 The evaluator ***shall examine*** the statement of TOE security assurance requirements to determine that either it includes an EAL as defined in CC Part 3 or appropriately justifies that it does not include an EAL.

397 If no EAL is included, the evaluator determines that the justification addresses why the statement of TOE assurance requirements contains no EAL. This justification may address the reason why it was impossible, undesirable or inappropriate to include an EAL, or it may address why it was impossible, undesirable or inappropriate to include particular components of the families that constitute EAL1 (ACM_CAP, ADO_IGS, ADV_FSP, ADV_RCR, AGD_ADM, AGD_USR, and ATE_IND).

ASE_REQ.1.4C

ASE_REQ.1-8 The evaluator ***shall examine*** the security requirements rationale to determine that it sufficiently justifies that the statement of TOE security assurance requirements is appropriate.

398 If the assurance requirements contain an EAL, the justification is allowed to address the choice of that EAL as a whole, rather than addressing all individual components of that EAL. If the assurance requirements contain augmented components to that EAL, the evaluator determines that each augmentation is individually justified. If the assurance requirements contain explicitly stated assurance requirements, the evaluator determines that the use of each explicitly stated assurance requirement is individually justified.

399 The evaluator determines that the security requirements rationale sufficiently justifies that the assurance requirements are sufficient given the statement of security environment and security objectives. For example, if defence against knowledgeable attackers is required, then it would be inappropriate to specify AVA_VLA.1 which is unlikely to detect other than obvious security weaknesses.

400 The justification may also include reasons such as:

a) the assurance requirements that appear in PPs that the ST claims conformance to;

b) specific requirements imposed by the scheme, national government, or other organisations;

c) assurance requirements that were dependencies from TOE security functional requirement;

d) assurance requirements of systems and/or products that are to be used in conjunction with the TOE;

e) consumer requirements.

401 An overview of the intent and goals of each EAL is provided in CC Part 3 section 6.2.

402 The evaluator is reminded that determining whether the assurance requirements are appropriate may be subjective and that the analysis of sufficiency of the justification should therefore not be overly rigorous.

403 If the assurance requirements do not contain an EAL, this work unit may be performed in conjunction with the ASE_REQ.1-7 work unit.

ASE_REQ.1.5C

ASE_REQ.1-9 The evaluator ***shall check*** that security requirements for the IT environment are identified, if appropriate.

404 If the ST does not contain security requirements for the IT environment, this work unit is not applicable and therefore considered to be satisfied.

405 The evaluator determines that any dependencies of the TOE on other IT in its environment to provide any security functionality in order for the TOE to achieve its security objectives are clearly identified in the ST as security requirements for the IT environment.

406 An example of a security requirement for the IT environment is a firewall that relies on an underlying operating system to provide authentication of administrators and permanent storage of audit data. In this case, the security requirements for the IT environment would contain components from the FAU and FIA classes.

407 Note that the security requirements for the IT environment can contain both functional and assurance requirements.

408 An example of a dependency on the IT environment is a software crypto-module, which periodically inspects its own code, and disables itself when the code has been tampered with. To allow for recovery, it has the requirement FPT_RCV.2 (automated recovery). As it cannot recover itself once it has disabled itself, this becomes a requirement on the IT environment. One of the dependencies of FPT_RCV.2 is AGD_ADM.1 (administrator guidance). This assurance requirement therefore becomes an assurance requirement for the IT environment.

409 The evaluator is reminded that where security requirements for the IT environment refer to the TSF, they refer to the security functions of the environment, rather than security functions of the TOE.

ASE_REQ.1.6C

ASE_REQ.1-10 The evaluator ***shall check*** that all operations on IT security requirements are identified.

Interp Note : Paragraphs 410-411 are replaced as a result of Interpretations 019 and 138.

410 ~~The permitted operations for CC Part 2 functional components are assignment, iteration, selection and refinement. The assignment and selection operations are permitted only where specifically indicated in a component. Iteration and refinement are permitted for all functional components.~~

411 ~~The permitted operations for CC Part 3 assurance components are iteration and refinement.~~

The permitted operations for CC Part 2 and Part 3 components are assignment, iteration, selection and refinement. The assignment and selection operations are permitted only where specifically indicated in a component. Iteration and refinement are permitted for all components.

412 The evaluator determines that all operations are identified in each component where such an operation is used. Identification can be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

ASE_REQ.1-11 The evaluator ***shall examine*** the statement of IT security requirements to determine that all assignment and selection operations are performed.

413 The evaluator determines that all assignments and selections in all components have either been completely performed (there are no choices left to be made in the component) or that is it appropriately justified that is not completely performed.

414 An example of not completely performing an operation is specifying a range of values when performing the assignment operation on the number of concurrent sessions that belong to the same user in FTA_MCS.1 (basic limitation on multiple concurrent sessions). An appropriate justification for this is that the value will be selected from the range of values by the administrator during TOE installation.

ASE_REQ.1-12 The evaluator ***shall examine*** the ST to determine that all operations are performed correctly.

415 The evaluator compares each statement with the element from which it is derived to determine that:

a) for an assignment, the values of the parameters or variables chosen comply with the indicated type required by the assignment;

b) for a selection, the selected item or items are one or more of the items indicated within the selection portion of the element. The evaluator also determines that the number of items chosen is appropriate for the requirement. Some requirements require a selection of just one item (e.g. FAU_GEN.1.1.b), in other cases multiple items (e.g. FDP_ITT.1.1 second operation) are acceptable.

c) for a refinement, the component is refined in such manner that a TOE meeting the refined requirement also meets the unrefined requirement. If the refined requirement exceeds this boundary it is considered to be an extended requirement.

Example: ADV_SPM.1.2C The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modelled.
Refinement: The TSP model need cover only access control.
If the access control policy is the only policy of the TSP this is a valid refinement. If there are also identification and authentication policies in the TSP, and the refinement is meant to state that only access control needs to be modeled, then this is not a valid refinement.

A special case of refinement is an editorial refinement, where a small change is made in a requirement, i.e. rephrasing a sentence due to adherence to proper English grammar. This change is not allowed to modify the meaning of the requirement in any way.

An example of an editorial refinement is FAU_ARP.1 with a single action. Instead of writing: “The TSF shall take *inform the operator* upon detection of a potential security violation” the ST author is allowed to write: “The TSF shall *inform the operator* upon detection of a potential security violation”.

The evaluator is reminded that editorial refinements have to be clearly identified (see work unit ASE_REQ.1-10).

d) for an iteration, that each iteration of a component is different from each other iteration of that component (at least one element of a component is different from the corresponding element of the other component), or that the component applies to a different part of the TOE.

ASE_REQ.1.7C

ASE_REQ.1-13 The evaluator ***shall examine*** the statement of IT security requirements to determine that dependencies required by the components used in the IT security requirements statement are satisfied.

416 Dependencies may be satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of TOE security requirements, or as a requirement that is asserted as being met by the IT environment of the TOE.

417 Although the CC provides support for dependency analysis by inclusion of dependency, this is not a justification that no other dependencies exist. An example of such other dependencies is an element that refers to “all objects” or “all subjects”, where a dependency could exist to a refinement in another element or set of elements where the objects or subjects are enumerated.

418 Dependencies of security requirements necessary in the IT environment should be stated and satisfied in the ST.

419 The evaluator is reminded that the CC does not require all dependencies to be satisfied: see the following work-unit.

ASE_REQ.1.8C

ASE_REQ.1-14 The evaluator ***shall examine*** the security requirements rationale to determine that an appropriate justification is given for each case where security requirement dependencies are not satisfied.

420 The evaluator determines that the justification explains why the dependency is unnecessary, given the identified security objectives.

421 The evaluator confirms that any non-satisfaction of a dependency does not prevent the set of security requirements adequately addressing the security objectives. This analysis is addressed by ASE_REQ.1.12C.

422 An example of an appropriate justification is when a software TOE has the security objective: “failed authentications shall be logged with user identity, time and date” and uses FAU_GEN.1 (audit data generation) as a functional requirement to satisfy this security objective. FAU_GEN.1 contains a dependency on FPT_STM.1 (reliable time stamps). As the TOE does not contain a clock mechanism, FPT_STM.1 is defined by the ST author as a requirement on the IT environment. The ST author indicates that this requirement will not be satisfied with the justification: “there are attacks possible on the time-stamping mechanism in this particular environment, the environment can therefore not deliver a reliable time-stamp. Yet, some threat agents are incapable of executing attacks against the time-stamping mechanisms, and some attacks by these threat agents may be analysed by logging time and date of their attacks.”

ASE_REQ.1.9C

ASE_REQ.1-15 The evaluator ***shall check*** that the ST includes a statement of the minimum strength of function level for the TOE security functional requirements, and that this level is either SOF-basic, SOF-medium or SOF-high.

423 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

424 The strength of cryptographic algorithms is outside the scope of the CC. Strength of function only applies to probabilistic or permutational mechanisms that are non-cryptographic. Therefore, where an ST contains a minimum SOF claim this claim does not apply to any cryptographic mechanisms with respect to a CC evaluation. Where such cryptographic mechanisms are included in a TOE the evaluator determines that the ST includes a clear statement that the assessment of algorithmic strength does not form part of the evaluation.

425 The TOE may contain multiple distinct domains, where the ST writer deems it to be more applicable to have a minimum strength of function level for each domain, rather than having one overall minimum strength of function level for the entire TOE. In this case it is allowed to partition the TOE security functional requirements in distinct sets, and have different minimum strength of function levels associated with each set.

426 An example of this is a distributed terminal system which has user terminals that are in a public space, and administrator terminals that are in a physically secure place. The authentication requirements for the user terminals have SOF-medium associated with them, and the authentication requirements for the administrative terminals have SOF-basic associated with them. Rather than stating that the TOE has a minimum strength of function level of SOF-basic, which might lead potential consumers of the TOE to believe that it would be relatively easy to successfully attack the authentication mechanisms on user terminals, the ST writer divides the TOE into a user domain and an administrative domain, partitions the TOE security functional requirements into sets belonging to those domains, assigns a minimum strength of function level of SOF-basic to the set belonging to the administrative domain, and assigns a minimum strength of function level of SOF-medium to the set belonging to the user domain.

ASE_REQ.1.10C

Interp Note : *Work unit ASE_REQ.1-16 is changed as a result of Interpretation 085.*

ASE_REQ.1-16 The evaluator ***shall check*** that the ST identifies any specific TOE security functional requirements for which an explicit strength of function is appropriate, together with the specific strength of function or metric as applicable.

427 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

Interp Note : *Paragraph 428 is changed as a result of Interpretation 085.*

428 The explicit strength of function claim can be either SOF-basic, SOF-medium, SOF-high, or a defined specific metric. Where a specific metric is used, the evaluator determines that these are appropriate for the type of functional requirement specified, and that the metric specified is evaluatable as a strength claim. This work unit refers to the case where an ST author requires to set specific SOF requirements (i.e. higher than the overall SOF claim of the ST) or by using a metric. A specific SOF claim for a TOE security functional requirement may be specified by a PP author. In the absence of any specific claim, the overall claim for the ST applies for all TOE security functional requirements stated in the ST. The evaluator should confirm the presence or absence of explicit SOF claims is consistent with other parts of the ST.

Interp Note : *The following paragraph is added as a result of Interpretation 085.*

An ST could potentially have varying specifications of SOF claims. There can be an overall SOF claim for an ST and within an ST the TOE security functional requirements could have a SOF claim specified for it.

429 Further guidance on appropriateness and suitability of strength of function metrics may be provided by the scheme.

ASE_REQ.1.11C

ASE_REQ.1-17 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the minimum strength of function level, together with any explicit strength of function claim, is consistent with the security objectives for the TOE.

430 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

431 The evaluator determines that the rationale takes into account details about the likely expertise, resources, and motivation of attackers as described in the statement of TOE security environment. For example, a claim of SOF-basic is inappropriate if the TOE is required to provide defence against attackers who possess a high attack potential.

432 The evaluator also determines that the rationale takes into account any specific strength-related properties of security objectives. The evaluator can use the tracings from requirements to objectives to determine that requirements that trace towards objectives with specific strength related properties, if appropriate, have a suitable strength of function claim associated with them.

ASE_REQ.1.12C

ASE_REQ.1-18 The evaluator ***shall examine*** the security requirements rationale to determine that the TOE security requirements are traced back to the security objectives for the TOE.

433 The evaluator determines that each TOE security functional requirement is traced back to at least one security objective for the TOE.

434 Failure to trace implies that either the security requirements rationale is incomplete, the security objectives are incomplete, or that the TOE security functional requirement has no useful purpose.

435 It is also allowed, but not mandatory, for some or all TOE security assurance requirements to trace back to security objectives for the TOE.

436 An example of a TOE security assurance requirement tracing back to a security objective for the TOE is an ST containing the threat “A user unwittingly discloses information by using a device thinking it to be the TOE” and the security objective for the TOE “The TOE shall be clearly labelled with its version number” to counter that threat. This security objective for the TOE can be achieved by satisfying ACM_CAP.1 and the ST author therefore traces ACM_CAP.1 back to that security objective for the TOE.

ASE_REQ.1-19 The evaluator ***shall examine*** the security requirements rationale to determine that the security requirements for the IT environment are traced back to the security objectives for the environment.

437 The evaluator determines that each functional security requirement for the IT environment is traced back to at least one security objective for the environment.

438 Failure to trace implies that either the security requirements rationale is incomplete, the security objectives for the environment are incomplete, or that the functional security requirement for the IT environment has no useful purpose.

439 It is also allowed, but not mandatory, for some or all security assurance requirements for the IT environment to trace back to security objectives for the environment.

Interp Note : Work unit ASE_REQ.1-20 is changed as a result of Interpretation 084.

ASE_REQ.1-20 The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the TOE it contains an appropriate justification that the TOE security requirements are suitable to meet that security objective for the TOE.

440 If no TOE security requirements trace back to the security objective for the TOE, this work unit fails.

441 The evaluator determines that the justification for a security objective for the TOE demonstrates that if all TOE security requirements that trace back to the objective are satisfied, the security objective for the TOE is achieved.

442 The evaluator also determines that each TOE security requirement that traces back to a security objective for the TOE, when satisfied, actually contributes to achieving the security objective.

443 Note that the tracings from TOE security requirements to security objectives for the TOE provided in the security requirements rationale may be a part of the justification, but do not constitute a justification by themselves.

ASE_REQ.1-21 The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the IT environment it contains an appropriate justification that the security requirements for the IT environment are suitable to meet that security objective for the IT environment.

444 If no security requirements for the IT environment trace back to the security objective for the IT environment, this work unit fails.

445 The evaluator determines that the justification for a security objective for the environment demonstrates that if all security requirements for the IT environment that trace back to the security objective for the IT environment are satisfied, the security objective for the IT environment is achieved.

446 The evaluator also determines that each security requirement for the IT environment that traces back to a security objective for the IT environment, when satisfied, actually contributes to achieving the security objective.

447 Note that the tracings from security requirements for the IT environment to security objectives for the IT environment provided in the security requirements

rationale may be a part of a justification, but do not constitute a justification by themselves.

ASE_REQ.1.13C

ASE_REQ.1-22 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the set of IT security requirements is internally consistent.

448 The evaluator determines that on all occasions where different IT security requirements apply to the same types of events, operations, data, tests to be performed etc., and these requirements might conflict, an appropriate justification is provided that this is not the case.

449 For example, if the ST contains requirements for individual accountability of users as well as requirements for user anonymity, it needs to be shown that these requirements do not conflict. This might involve showing that none of the auditable events requiring individual user accountability relate to operations for which user anonymity is required.

450 For guidance on consistency analysis see Annex B.3.

ASE_REQ.1-23 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the set of IT security requirements together forms a mutually supportive whole.

451 This work unit builds on the determination performed in work units ASE_REQ.1-18 and ASE_REQ.1-19, which examine the tracing from IT security requirements to security objectives and work units ASE_REQ.1-20 and ASE_REQ.1-21 which examine whether the IT security requirements are suitable to meet the security objectives. This work unit requires the evaluator to consider the possibility that a security objective might in fact not be achieved because of lack of support from other IT security requirements.

452 This work unit also builds on the dependency analysis addressed by previous work units, because if functional requirement A has a dependency on functional requirement B, B supports A by definition.

453 The evaluator determines that the security requirements rationale demonstrates that functional requirements support each other where necessary, even when no dependency between these requirements is indicated. This demonstration should address security functional requirements that:

a) prevent bypass of other security functional requirements, such as FPT_RVM.1;

b) prevent tampering with other security functional requirements, such as FPT_SEP;

c) prevent de-activation of other security functional requirements, such as FMT_MOF.1;

d) enable detection of attacks aimed at defeating other security functional requirements, such as components of the FAU class.

454 The evaluator takes the performed operations into account in his analysis to determine whether they affect the mutual support between the requirements.

4.4.6.3.2 Action ASE_REQ.1.2E

ASE_REQ.1-24 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is coherent.

455 The statement of IT security requirements is coherent if the text and structure of the statement are understandable by its target audience (i.e. evaluators and consumers).

ASE_REQ.1-25 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is complete.

456 This work unit draws on the results from the work units required by ASE_REQ.1.1E and ASE_SRE.1.1E, and in particular the evaluator's examination of the security requirements rationale.

457 The statement of security requirements is complete if all operations on requirements have been completed, and the evaluator judges the security requirements to be sufficient to ensure that all security objectives for the TOE are satisfied.

ASE_REQ.1-26 The evaluator ***shall examine*** the statement of IT security requirements to determine that it is internally consistent.

458 This work unit draws on the results from the work units required by ASE_REQ.1.1E and ASE_SRE.1.1E, and in particular the evaluator's examination of the security requirements rationale.

459 The statement of security requirements is internally consistent if the evaluator determines that no security requirement conflicts with any other security requirement, such that a security objective will not be fully satisfied.

460 For guidance on consistency analysis see Annex B.3.

4.4.7 Evaluation of explicitly stated IT security requirements (ASE_SRE.1)

4.4.7.1 Objectives

461 The objective of this sub-activity is to determine whether the security functional requirements or security assurance requirements that are stated without reference to the CC are appropriate and adequate.

4.4.7.2 Application Notes

462 This section is only applicable if the ST contains IT security requirements that are explicitly stated without reference to either CC Part 2 or CC Part 3. If this is not the case, all work units in this section are not applicable, and therefore considered to be satisfied.

463 The ASE_SRE requirements do not replace the ASE_REQ requirements, but are additional to them. This means that IT security requirements that are explicitly stated without reference to either CC Part 2 or CC Part 3 must be evaluated with the ASE_SRE criteria, and also, in combination with all other security requirements, with the ASE_REQ criteria.

4.4.7.3 Input

464 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.7.4 Evaluator actions

465 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_SRE.1.1E;

b) ASE_SRE.1.2E.

4.4.7.4.1 Action ASE_SRE.1.1E

ASE_SRE.1.1C

ASE_SRE.1-1 The evaluator ***shall check*** that the statement of the IT security requirements identifies all TOE security requirements that are explicitly stated without reference to the CC.

466 Any TOE security functional requirements that are not specified using CC Part 2 functional components are required to be clearly identified as such. Similarly, any TOE security assurance requirements that are not specified using CC Part 3 assurance components are also required to be clearly identified as such.

ASE_SRE.1.2C

ASE_SRE.1-2 The evaluator ***shall check*** that the statement of IT security requirements identifies all security requirements for the IT environment that are explicitly stated without reference to the CC.

467 Any security functional requirements for the IT environment that are not specified using CC Part 2 functional components are required to be clearly identified as such. Similarly, any security assurance requirements for the IT environment that

are not specified using CC Part 3 assurance components are also required to be clearly identified as such.

ASE_SRE.1.3C

ASE_SRE.1-3 The evaluator ***shall examine*** the security requirements rationale to determine that it appropriately justifies why each explicitly stated IT security requirement had to be explicitly stated.

468 The evaluator determines for each explicitly stated IT security requirement that the justification explains why existing functional or assurance components (from CC Part 2 and CC Part 3, respectively) could not be used to express the explicitly stated security requirement in question. The evaluator takes the possibility of performing operations (i.e. assignment, iteration, selection or refinement) on these existing components into account in this determination.

ASE_SRE.1.4C

ASE_SRE.1-4 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that the requirement uses the CC requirements components, families and classes as a model for presentation.

469 The evaluator determines that explicitly stated IT security requirements are presented in the same style as CC Part 2 or CC Part 3 components and to a comparable level of detail. The evaluator also determines that the functional requirements are broken down into individual functional elements and that the assurance requirements specify the developer action, content and presentation of evidence, and evaluator action elements.

ASE_SRE.1.5C

ASE_SRE.1-5 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that it is measurable and states objective evaluation requirements, such that compliance or noncompliance of a TOE can be determined and systematically demonstrated.

470 The evaluator determines that functional requirements are stated in such a way that they are testable, and traceable through the appropriate TSF representations. The evaluator also determines that assurance requirements avoid the need for subjective evaluator judgement.

Interp Note : The following paragraph is added as a result of Interpretation 064.

The existing CC functional and assurance requirements are to be used as models for compliance with this requirement.

ASE_SRE.1.6C

ASE_SRE.1-6 The evaluator ***shall examine*** each explicitly stated IT security requirement to determine that it is clearly and unambiguously expressed.

Interp Note : *The following paragraph is added as a result of Interpretation 064.*

The existing CC functional and assurance requirements are to be used as models for compliance with this requirement.

ASE_SRE.1.7C

ASE_SRE.1-7 The evaluator ***shall examine*** the security requirements rationale to determine that it demonstrates that the assurance requirements are applicable and appropriate to support any explicitly stated TOE security functional requirements.

471 The evaluator determines whether application of the specified assurance requirements will yield a meaningful evaluation result for each explicitly stated security functional requirement, or whether other assurance requirements should have been specified. For example, an explicitly stated functional requirement may imply the need for particular documentary evidence (such as a TSP model), depth of testing, or analysis (such as strength of TOE security functions analysis or covert channel analysis).

4.4.7.4.2 Action ASE_SRE.1.2E

ASE_SRE.1-8 The evaluator ***shall examine*** the statement of IT security requirements to determine that all of the dependencies of any explicitly stated IT security requirement have been identified.

472 The evaluator confirms that no applicable dependencies have been overlooked by the ST author.

473 Examples of possible dependencies are: components of the FAU class if an explicitly stated functional requirement mentions auditing and ADV_IMP if an explicitly stated assurance requirement mentions the source code or implementation representation of the TOE.

4.4.8 Evaluation of TOE summary specification (ASE_TSS.1)

4.4.8.1 Objectives

474 The objective of this sub-activity is to determine whether the TOE summary specification provides a clear and consistent high-level definition of the security functions and assurance measures, and that these satisfy the specified TOE security requirements.

4.4.8.2 Input

475 The evaluation evidence for this sub-activity is:

a) the ST.

4.4.8.3 Evaluator actions

476 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ASE_TSS.1.1E;

b) ASE_TSS.1.2E.

4.4.8.3.1 Action ASE_TSS.1.1E

ASE_TSS.1.1C

ASE_TSS.1-1 The evaluator ***shall check*** that the TOE summary specification describes the IT security functions and assurance measures of the TOE.

477 The evaluator determines that the TOE summary specification provides a high-level definition of the security functions claimed to meet the TOE security functional requirements, and of the assurance measures claimed to meet the TOE security assurance requirements.

478 The assurance measures can be explicitly stated, or defined by reference to the documents that satisfy the security assurance requirements (e.g. relevant quality plans, life cycle plans, management plans).

ASE_TSS.1.2C

ASE_TSS.1-2 The evaluator ***shall check*** the TOE summary specification to determine that each IT security function is traced to at least one TOE security functional requirement.

479 Failure to trace implies that either the TOE summary specification is incomplete, the TOE security functional requirements are incomplete, or the IT security function has no useful purpose.

ASE_TSS.1.3C

ASE_TSS.1-3 The evaluator ***shall examine*** each IT security function to determine that it is described in an informal style to a level of detail necessary for understanding its intent.

480 In some cases, an IT security function may provide no more detail than is provided in the corresponding TOE security functional requirement or requirements. In others, the ST author may have included TOE-specific details, for example using TOE-specific terminology in place of generic terms such as ‘security attribute’.

481 Note that a semi-formal or formal style of describing IT security functions is not allowed here, unless accompanied by an informal style description of the same functions. The goal here is to understand the intent of the function, rather than determining properties such as completeness or correctness of the functions.

ASE_TSS.1.4C

ASE_TSS.1-4 The evaluator ***shall examine*** the TOE summary specification to determine that all references to security mechanisms in the ST are traced back to IT security functions.

482 References to security mechanisms are optional in an ST but may (for example) be appropriate where there is a requirement to implement particular protocols or algorithms (e.g. specified password generation or encryption algorithms). If the ST contains no references to security mechanisms, this work unit is not applicable and is therefore considered to be satisfied.

483 The evaluator determines that each security mechanism that the ST refers to is traced back to at least one IT security function.

484 Failure to trace implies that either the TOE summary specification is incomplete or the security mechanism has no useful purpose.

ASE_TSS.1.5C

ASE_TSS.1-5 The evaluator ***shall examine*** the TOE summary specification rationale to determine that for each TOE security functional requirement it contains an appropriate justification that the IT security functions are suitable to meet that TOE security functional requirement.

485 If no IT security functions trace back to the TOE security functional requirement, this work unit fails.

486 The evaluator determines that the justification for a TOE security functional requirement demonstrates that if all IT security functions that trace back to that requirement are implemented, the TOE security functional requirement is met.

487 The evaluator also determines that each IT security function that traces back to a TOE security functional requirement, when implemented, actually contributes to meeting that requirement.

488 Note that the tracings from IT security functions to TOE security functional requirements provided in the TOE summary specification may be a part of a justification, but do not constitute a justification by themselves.

ASE_TSS.1-6 The evaluator ***shall examine*** the TOE summary specification rationale to determine that the strength of function claims for the IT security functions are consistent with the strength of functions for the TOE security functional requirements.

489 This work unit draws on the results of the ASE_TSS.1-10 work unit.

Interp Note : The following paragraph is changed as a result of Interpretation 127.

490 The evaluator determines that for each IT security function for which a strength of function claim is appropriate, the TOE summary specification rationale

demonstrates that this claim is adequate for all TOE security functional requirements that it traces back to.

491 Usually adequacy means that the strength of function claim of the IT security function is equal to or higher than the strength of function of all TOE security functional requirements that it traces to, but exceptions are possible. An example of such an exception is the case where multiple low strength functions are used sequentially to implement a medium strength authentication requirement for authentication (e.g. biometry and a PIN).

ASE_TSS.1.6C

ASE_TSS.1-7 The evaluator ***shall examine*** the TOE summary specification rationale to determine that it demonstrates that the combination of the specified IT security functions work together so as to satisfy the TOE security functional requirements.

492 This work unit builds on the determination of mutual support performed on the TOE security functional requirements in work unit ASE_REQ.1-23. The evaluator's analysis here should assess the impact of additional information included in the IT security functions to determine that the inclusion of such information introduces no potential security weaknesses, such as possibilities to bypass, tamper with, or deactivate other IT security functions.

ASE_TSS.1.7C

ASE_TSS.1-8 The evaluator ***shall check*** the TOE summary specification to determine that each assurance measure is traced to at least one TOE security assurance requirement.

493 Failure to trace implies that either the TOE summary specification or the statement of TOE security assurance requirements is incomplete, or that the assurance measure has no useful purpose.

ASE_TSS.1.8C

ASE_TSS.1-9 The evaluator ***shall examine*** the TOE summary specification rationale to determine that for each TOE security assurance requirement it contains an appropriate justification that the assurance measures meet that TOE security assurance requirement.

494 If no assurance measures trace back to the TOE security assurance requirement, this work unit fails.

495 The evaluator determines that the justification for a TOE security assurance requirement demonstrates that if all assurance measures that trace back to that requirement are implemented, the TOE security assurance requirement is met.

496 The evaluator also determines that each assurance measure that traces back to a TOE security assurance requirement, when implemented, actually contributes to meeting that requirement.

497 An assurance measure describes how the developer will address the assurance requirements. The aim of this work unit is to determine that the specified assurance measures are appropriate to satisfy the assurance requirements.

498 Note that the tracings from assurance measures to TOE security assurance requirements provided in the TOE summary specification may be a part of a justification, but do not constitute a justification by themselves.

ASE_TSS.1.9C

ASE_TSS.1-10 The evaluator ***shall check*** that the TOE summary specification identifies all IT security functions that are realised by a probabilistic or permutational mechanisms.

499 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

500 This work unit might be revisited after analysis of other evaluation evidence identifies permutational or probabilistic mechanisms that are not identified as such in the ST.

ASE_TSS.1.10C

ASE_TSS.1-11 The evaluator ***shall check*** that, for each IT security function for which it is appropriate, the TOE summary specification states the strength of function claim either as a specific metric or as SOF-basic, SOF-medium or SOF-high.

501 If the TOE security assurance requirements do not include AVA_SOF.1, this work unit is not applicable and is therefore considered to be satisfied.

4.4.8.3.2 Action ASE_TSS.1.2E

ASE_TSS.1-12 The evaluator ***shall examine*** the TOE summary specification to determine that it is complete.

502 The TOE summary specification is complete if the evaluator judges the IT security functions and assurance measures to be sufficient to ensure that all specified TOE security requirements are satisfied. This work unit should be performed in conjunction with the ASE_TSS.1-5 and ASE_TSS.1-9 work units.

ASE_TSS.1-13 The evaluator ***shall examine*** the TOE summary specification to determine that it is coherent.

503 The TOE summary specification is coherent if its text and structure are understandable by its target audience (i.e. evaluators and developers).

ASE_TSS.1-14 The evaluator ***shall examine*** the TOE summary specification to determine that it is internally consistent.

504 The TOE summary specification is internally consistent if the evaluator determines there is no conflict between IT security functions or assurance measures, such that a security requirement for the TOE will not be fully satisfied.

505 For guidance on consistency analysis see Annex B.3.

Chapter 5

EAL1 evaluation

5.1 Introduction

506 EAL1 provides a basic level of assurance. The security functions are analysed using a functional specification and guidance documentation to understand the security behaviour. Independent testing of a subset of the TOE security functions is performed.

5.2 Objectives

507 The objective of this chapter is to define the minimal evaluation effort for achieving an EAL1 evaluation and to provide guidance on ways and means of accomplishing the evaluation.

5.3 EAL1 evaluation relationships

508 An EAL1 evaluation covers the following:

a) evaluation input task (Chapter 2);

b) EAL1 evaluation activities comprising the following:

   1) evaluation of the ST (Chapter 4);

   2) evaluation of the configuration management (Section 5.4);

   3) evaluation of the delivery and operation documents (Section 5.5);

   4) evaluation of the development documents (Section 5.6);

   5) evaluation of the guidance documents (Section 5.7);

   6) testing (Section 5.8);

c) evaluation output task (Chapter 2).

509 The evaluation activities are derived from the EAL1 assurance requirements contained in the CC Part 3.

510 The ST evaluation is started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities.

511 The sub-activities comprising an EAL1 evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator.

512 For guidance on dependencies see Annex B.4.

5.4 Configuration management activity

513 The purpose of the configuration management activity is to assist the consumer in identifying the evaluated TOE.

514 The configuration management activity at EAL1 contains a sub-activity related to the following component:

a) ACM_CAP.1.

5.4.1 Evaluation of CM capabilities (ACM_CAP.1)

5.4.1.1 Objectives

515 The objectives of this sub-activity are to determine whether the developer has clearly identified the TOE.

5.4.1.2 Input

516 The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing.

5.4.1.3 Evaluator action

517 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_CAP.1.1E.

5.4.1.3.1 Action ACM_CAP.1.1E

ACM_CAP.1.1C

1:ACM_CAP.1-1 The evaluator ***shall check*** that the version of the TOE provided for evaluation is uniquely referenced.

518 For this assurance component there is no requirement for the developer to use a CM system, beyond unique referencing. As a result the evaluator is able to verify the uniqueness of a TOE version only by checking that other versions of the TOE available for purchase do not possess the same reference. In evaluations where a CM system was provided in excess of the CC requirements, the evaluator could validate the uniqueness of the reference by checking the configuration list. Evidence that the version provided for evaluation is uniquely referenced may be incomplete if only one version is examined during the evaluation, and the evaluator should look for a referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates). However, the absence of any reference will normally lead to a fail verdict against this requirement unless the evaluator is confident that the TOE can be uniquely identified.

519 The evaluator should seek to examine more than one version of the TOE (e.g. during rework following discovery of a vulnerability), to check that the two versions are referenced differently.

ACM_CAP.1.2C

1:ACM_CAP.1-2 The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

520 The evaluator should ensure that the TOE contains a unique reference such that it is possible to distinguish different versions of the TOE. This could be achieved through labelled packaging or media, or by a label displayed by the operational TOE. This is to ensure that it would be possible for consumers to identify the TOE (e.g. at the point of purchase or use).

521 The TOE may provide a method by which it can be easily identified. For example, a software TOE may display its name and version number during the start up routine, or in response to a command line entry. A hardware or firmware TOE may be identified by a part number physically stamped on the TOE.

1:ACM_CAP.1-3 The evaluator ***shall check*** that the TOE references used are consistent.

522 If the TOE is labelled more than once then the labels have to be consistent. For example, it should be possible to relate any labelled guidance documentation supplied as part of the TOE to the evaluated operational TOE. This ensures that consumers can be confident that they have purchased the evaluated version of the TOE, that they have installed this version, and that they have the correct version of the guidance to operate the TOE in accordance with its ST.

523 The evaluator also verifies that the TOE reference is consistent with the ST.

524 For guidance on consistency analysis see Annex B.3.

5.5 Delivery and operation activity

525 The purpose of the delivery and operation activity is to judge the adequacy of the documentation of the procedures used to ensure that the TOE is installed, generated, and started in the same way the developer intended it to be.

526 The delivery and operation activity at EAL1 contains a sub-activity related to the following component:

a) ADO_IGS.1.

5.5.1 Evaluation of installation, generation and start-up (ADO_IGS.1)

5.5.1.1 Objectives

527 The objective of this sub-activity is to determine whether the procedures and steps for the secure installation, generation, and start-up of the TOE have been documented and result in a secure configuration.

5.5.1.2 Input

528 The evaluation evidence for this sub-activity is:

a) the administrator guidance;

b) the secure installation, generation, and start-up procedures;

c) the TOE suitable for testing.

5.5.1.3 Application notes

529 The installation, generation, and start-up procedures refer to all installation, generation, and start-up procedures, regardless of whether they are performed at the user's site or at the development site that are necessary to progress the TOE to the secure configuration as described in the ST.

5.5.1.4 Evaluator actions

530 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_IGS.1.1E;

b) ADO_IGS.1.2E.

5.5.1.4.1 Action ADO_IGS.1.1E

ADO_IGS.1.1C

1:ADO_IGS.1-1 The evaluator ***shall check*** that the procedures necessary for the secure installation, generation and start-up of the TOE have been provided.

531 If it is not anticipated that the installation, generation, and start-up procedures will or can be re-applied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

5.5.1.4.2 Action ADO_IGS.1.2E

1:ADO_IGS.1-2 The evaluator ***shall examine*** the provided installation, generation, and start-up procedures to determine that they describe the steps necessary for secure installation, generation, and start-up of the TOE.

532 If it is not anticipated that the installation, generation, and start-up procedures will or can be re-applied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

533 The installation, generation, and start-up procedures may provide detailed information about the following:

a) changing the installation specific security characteristics of entities under the control of the TSF;

b) handling exceptions and problems;

c) minimum system requirements for secure installation if applicable.

534 In order to confirm that the installation, generation, and start-up procedures result in a secure configuration, the evaluator may follow the developer's procedures and may perform the activities that customers are usually expected to perform to install, generate, and start-up the TOE (if applicable to the TOE), using the supplied guidance documentation only. This work unit might be performed in conjunction with the 1:ATE_IND.1-2 work unit.

5.6 Development activity

535 The purpose of the development activity is to assess the design documentation in terms of its adequacy to understand how the TSF provides the security functions of the TOE. This understanding is achieved through examination of a functional specification (which describes the external interfaces of the TOE) and a representation correspondence (which maps the functional specification to TOE summary specification in order to ensure consistency).

536 The development activity at EAL1 contains sub-activities related to the following components:

a) ADV_FSP.1;

b) ADV_RCR.1.

5.6.1 Application notes

537 The CC requirements for design documentation are levelled by formality. The CC considers a document's degree of formality (that is, whether it is informal, semiformal or formal) to be hierarchical. An informal document is one that is expressed in a natural language. The methodology does not dictate the specific language that must be used; that issue is left for the scheme. The following paragraphs differentiate the contents of the different informal documents.

538 An informal functional specification comprises a description the security functions (at a level similar to that of the TOE summary specification) and a description of the externally-visible interfaces to the TSF. For example, if an operating system presents the user with a means of self-identification, of creating files, of modifying or deleting files, of setting permissions defining what other users may access files, and of communicating with remote machines, its functional specification would contain descriptions of each of these functions. If there are also audit functions that detect and record the occurrences of such events, descriptions of these audit functions would also be expected to be part of the functional specification; while these functions are technically not directly invoked by the user at the external interface, they certainly are affected by what occurs at the user's external interface.

539 Informality of the demonstration of correspondence need not be in a prose form; a simple two-dimensional mapping may be sufficient. For example, a matrix with modules listed along one axis and subsystems listed along the other, with the cells identifying the correspondence of the two, would serve to provide an adequate informal correspondence between the high-level design and the low-level design.

5.6.2 Evaluation of functional specification (ADV_FSP.1)

5.6.2.1 Objectives

540 The objective of this sub-activity is to determine whether the developer has provided an adequate description of the security functions of the TOE and whether

the security functions provided by the TOE are sufficient to satisfy the security functional requirements of the ST.

5.6.2.2 Input

541 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance.

5.6.2.3 Evaluator actions

542 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_FSP.1.1E;

b) ADV_FSP.1.2E.

5.6.2.3.1 Action ADV_FSP.1.1E

ADV_FSP.1.1C

1:ADV_FSP.1-1 The evaluator ***shall examine*** the functional specification to determine that it contains all necessary informal explanatory text.

543 If the entire functional specification is informal, this work unit is not applicable and is therefore considered to be satisfied.

544 Supporting narrative descriptions are necessary for those portions of the functional specification that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_FSP.1.2C

1:ADV_FSP.1-2 The evaluator ***shall examine*** the functional specification to determine that it is internally consistent.

545 The evaluator validates the functional specification by ensuring that the descriptions of the interfaces making up the TSFI are consistent with the descriptions of the functions of the TSF

546 For guidance on consistency analysis see Annex B.3.

ADV_FSP.1.3C

1:ADV_FSP.1-3 The evaluator ***shall examine*** the functional specification to determine that it identifies all of the external TOE security function interfaces.

547 The term *external* refers to that which is visible to the user. External interfaces to the TOE are either direct interfaces to the TSF or interfaces to non-TSF portions of the TOE. However, these non-TSF interfaces might have eventual access to the TSF. These external interfaces that directly or indirectly access the TSF collectively make up the TOE security function interface (TSFI). Figure 5.1 shows a TOE with TSF (shaded) portions and non-TSF (empty) portions. This TOE has three external interfaces: interface *c* is a direct interface to the TSF; interface *b* is an indirect interface to the TSF; and interface *a* is an interface to non-TSF portions of the TOE. Therefore, interfaces *b* and *c* make up the TFSI.

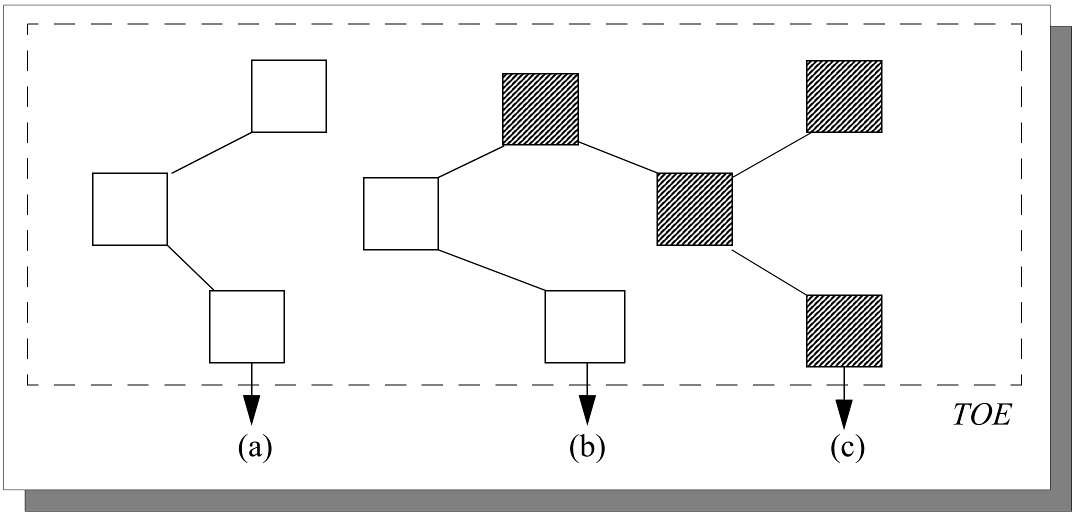

Figure 5.1 TSF Interfaces

548 It should be noted that all security functions reflected in the functional requirements of CC Part 2 (or in extended components thereof) will have some sort of externally-visible manifestation. While not all of these are necessarily interfaces from which the security function can be tested, they are all externally-visible to some extent and must therefore be included in the functional specification.

549 For guidance on determining the TOE boundary see Annex B.6.

1:ADV_FSP.1-4 The evaluator ***shall examine*** the functional specification to determine that it describes all of the external TOE security function interfaces.

550 For a TOE that has no threat of malicious users (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are rightfully excluded from its ST), the only interfaces that are

described in the functional specification (and expanded upon in the other TSF representation descriptions) are those to and from the TSF. The absence of FPT_PHP, FPT_RVM, and FPT_SEP presumes there is no concern for any sort of bypassing of the security features; therefore, there is no concern with any possible impact that other interfaces might have on the TSF.

551 On the other hand, if the TOE has a threat of malicious users or bypass (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are included in its ST), all external interfaces are described in the functional specification, but only to the extent that the effect of each is made clear: interfaces to the security functions (i.e. interfaces *b* and *c* in Figure 5.1) are completely described, while other interfaces are described only to the extent that it is clear that the TSF is inaccessible through the interface (i.e. that the interface is of type *a*, rather than *b* in Figure 5.1). The inclusion of FPT_PHP, FPT_RVM, and FPT_SEP implies a concern that all interfaces might have some effect upon the TSF. Because each external interface is a potential TSF interface, the functional specification must contain a description of each interface in sufficient detail so that an evaluator can determine whether the interface is security relevant.

552 Some architectures lend themselves to readily provide this interface description in sufficient detail for groups of external interfaces. For example, a kernel architecture is such that all calls to the operating system are handled by kernel programs; any calls that might violate the TSP must be called by a program with the privilege to do so. All programs that execute with privilege must be included in the functional specification. Any program external to the kernel that executes without privilege is incapable of affecting the TSP (i.e. such programs are interfaces of type *a*, rather than *b* in Figure 5.1) and may, therefore, be excluded from the functional specification. It is worth noting that, while the evaluator's understanding of the interface description can be expedited in cases where there is a kernel architecture, such an architecture is not necessary.

1:ADV_FSP.1-5 The evaluator ***shall examine*** the presentation of the TSFI to determine that it adequately and correctly describes the behaviour of the TOE at each external interface describing effects, exceptions and error messages.

553 In order to assess the adequacy and correctness of an interface's presentation, the evaluator uses the functional specification, the TOE summary specification of the ST, and the user and administrator guidance to assess the following factors:

a) All security relevant user input parameters (or a characterisation of those parameters) should be identified. For completeness, parameters outside of direct user control should be identified if they are usable by administrators.

b) All security relevant behaviour described in the reviewed guidance should be reflected in the description of semantics in the functional specification. This should include an identification of the behaviour in terms of events and the effect of each event. For example, if an operating system provides a rich file system interface, where it provides a different error code for each reason why a file is not opened upon request (e.g. access denied, no such file, file is in use by another user, user is not authorised to open the file after 5pm, etc.), the functional specification should explain that a file is either opened upon request, or else that an error code is returned. (While the functional specification may enumerate all these different reasons for errors, it need not provide such detail.) The description of the semantics should include how the security requirements apply to the interface (e.g. whether the use of the interface is an auditable event and, if so, the information that can be recorded).

c) All interfaces are described for all possible modes of operation. If the TSF provides the notion of privilege, the description of the interface should explain how the interface behaves in the presence or absence of privilege.

d) The information contained in the descriptions of the security relevant parameters and syntax of the interface should be consistent across all documentation.

554 Verification of the above is done by reviewing the functional specification and the TOE summary specification of the ST, as well as the user and administrator guidance provided by the developer. For example, if the TOE were an operating system and its underlying hardware, the evaluator would look for discussions of user-accessible programs, descriptions of protocols used to direct the activities of programs, descriptions of user-accessible databases used to direct the activities of programs, and for user interfaces (e.g. commands, application program interfaces) as applicable to the TOE under evaluation; the evaluator would also ensure that the processor instruction set is described.

555 This review might be iterative, such that the evaluator would not discover the functional specification to be incomplete until the design, source code, or other evidence is examined and found to contain parameters or error messages that have been omitted from the functional specification.

ADV_FSP.1.4C

1:ADV_FSP.1-6 The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

556 In order to assess the completeness of the TSF representation, the evaluator consults the TOE summary specification of the ST, the user guidance, and the administrator guidance. None of these should describe security functions that are absent from the TSF presentation of the functional specification.

5.6.2.3.2 Action ADV_FSP.1.2E

1:ADV_FSP.1-7 The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the TOE security functional requirements.

557 To ensure that all ST security functional requirements are covered by the functional specification, the evaluator may construct a map between the TOE summary specification and the functional specification. Such a map might be already provided by the developer as evidence for meeting the correspondence

(ADV_RCR.*) requirements, in which case the evaluator need only verify the completeness of this mapping, ensuring that all security functional requirements are mapped onto applicable TSFI presentations in the functional specification.

1:ADV_FSP.1-8 The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the TOE security functional requirements.

558 For each interface to a security function with specific characteristics, the detailed information in the functional specification must be exactly as it is specified in the ST. For example, if the ST contains user authentication requirements that the password length must be eight characters, the TOE must have eight-character passwords; if the functional specification describes six-character fixed length passwords, the functional specification would not be an accurate instantiation of the requirements.

559 For each interface in the functional specification that operates on a controlled resource, the evaluator determines whether it returns an error code that indicates a possible failure due to enforcement of one of the security requirements; if no error code is returned, the evaluator determines whether an error code should be returned. For example, an operating system might present an interface to OPEN a controlled object. The description of this interface may include an error code that indicates that access was not authorised to the object. If such an error code does not exist, the evaluator should confirm that this is appropriate (because, perhaps, access mediation is performed on READs and WRITEs, rather than on OPENs).

5.6.3 Evaluation of representation correspondence (ADV_RCR.1)

5.6.3.1 Objectives

560 The objective of this sub-activity is to determine whether the developer has correctly and completely implemented the requirements of the ST in the functional specification.

5.6.3.2 Input

561 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the correspondence analysis between the TOE summary specification and the functional specification.

5.6.3.3 Evaluator actions

562 This sub-activity comprises one CC Part 3 evaluator action element:

a) ADV_RCR.1.1E.

5.6.3.3.1 Action ADV_RCR.1.1E

ADV_RCR.1.1C

1:ADV_RCR.1-1 The evaluator ***shall examine*** the correspondence analysis between the TOE summary specification and the functional specification to determine that the functional specification is a correct and complete representation of the TOE security functions.

563 The evaluator's goal in this work unit is to determine that all security functions identified in the TOE summary specification are represented in the functional specification and that they are represented accurately.

564 The evaluator reviews the correspondence between the TOE security functions of the TOE summary specification and the functional specification. The evaluator looks for consistency and accuracy in the correspondence. Where the correspondence analysis indicates a relationship between a security function of the TOE summary specification and an interface description in the functional specification, the evaluator verifies that the security functionality of both are the same. If the security functions of the TOE summary specification are correctly and completely present in the corresponding interface, this work unit will be satisfied.

565 This work unit may be done in conjunction with work units 1:ADV_FSP.1-7 and 1:ADV_FSP.1-8.

5.7 Guidance documents activity

566 The purpose of the guidance document activity is to judge the adequacy of the documentation describing how to use the operational TOE. Such documentation includes both that aimed at trusted administrators and non-administrator users whose incorrect actions could adversely affect the security of the TOE, as well as that aimed at untrusted users whose incorrect actions could adversely affect the security of their own data.

567 The guidance documents activity at EAL1 contains sub-activities related to the following components:

a) AGD_ADM.1;

b) AGD_USR.1.

5.7.1 Application notes

568 The guidance documents activity applies to those functions and interfaces which are related to the security of the TOE. The secure configuration of the TOE is described in the ST.

5.7.2 Evaluation of administrator guidance (AGD_ADM.1)

5.7.2.1 Objectives

569 The objective of this sub-activity is to determine whether the administrator guidance describes how to administer the TOE in a secure manner.

5.7.2.2 Application notes

570 The term *administrator* is used to indicate a human user who is trusted to perform security critical operations within the TOE, such as setting TOE configuration parameters. The operations may affect the enforcement of the TSP, and the administrator therefore possesses specific privileges necessary to perform those operations. The role of the administrator(s) has to be clearly distinguished from the role of non-administrative users of the TOE.

571 There may be different administrator roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF such as auditor, administrator, or daily-management. Each role can encompass an extensive set of capabilities, or can be a single one. The capabilities of these roles and their associated privileges are described in the FMT class. Different administrator roles and groups should be taken into consideration by the administrator guidance.

5.7.2.3 Input

572 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures.

5.7.2.4 Evaluator actions

573 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_ADM.1.1E.

5.7.2.4.1 Action AGD_ADM.1.1E

AGD_ADM.1.1C

1:AGD_ADM.1-1 The evaluator ***shall examine*** the administrator guidance to determine that it describes the administrative security functions and interfaces available to the administrator of the TOE.

574 The administrator guidance should contain an overview of the security functionality that is visible at the administrator interfaces.

575 The administrator guidance should identify and describe the purpose, behaviour, and interrelationships of the administrator security interfaces and functions.

576 For each administrator security interface and function, the administrator guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system calls, menu selection, command button);

b) describe the parameters to be set by the administrator, their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_ADM.1.2C

1:AGD_ADM.1-2 The evaluator ***shall examine*** the administrator guidance to determine that it describes how to administer the TOE in a secure manner.

577 The administrator guidance describes how to operate the TOE according to the TSP in an IT environment that is consistent with the one described in the ST.

AGD_ADM.1.3C

1:AGD_ADM.1-3 The evaluator ***shall examine*** the administrator guidance to determine that it contains warnings about functions and privileges that should be controlled in a secure processing environment.

578 The configuration of the TOE may allow users to have dissimilar privileges to make use of the different functions of the TOE. This means that some users may be authorised to perform certain functions while other users may not be so authorised. These functions and privileges should be described by the administrator guidance.

579 The administrator guidance identifies the functions and privileges that must be controlled, the types of controls required for them, and the reasons for such controls. Warnings address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_ADM.1.4C

1:AGD_ADM.1-4 The evaluator ***shall examine*** the administrator guidance to determine that it describes all assumptions regarding user behaviour that are relevant to the secure operation of the TOE.

580 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the information that is of concern to the secure operation of the TOE need be included in the administrator guidance.

581 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

AGD_ADM.1.5C

1:AGD_ADM.1-5 The evaluator ***shall examine*** the administrator guidance to determine that it describes all security parameters under the control of the administrator indicating secure values as appropriate.

582 For each security parameter, the administrator guidance should describe the purpose of the parameter, the valid and default values of the parameter, and secure and insecure use settings of such parameters, both individually or in combination.

AGD_ADM.1.6C

1:AGD_ADM.1-6 The evaluator ***shall examine*** the administrator guidance to determine that it describes each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

583 All types of security-relevant events are detailed, such that an administrator knows what events may occur and what action (if any) the administrator may have to take in order to maintain security. Security-relevant events that may occur during operation of the TOE (e.g. audit trail overflow, system crash, updates to user records, such as when a user account is removed when the user leaves the

organisation) are adequately defined to allow administrator intervention to maintain secure operation.

AGD_ADM.1.7C

1:AGD_ADM.1-7 The evaluator ***shall examine*** the administrator guidance to determine that it is consistent with all other documents supplied for evaluation.

584 The ST in particular may contain detailed information on any warnings to the TOE administrators with regard to the TOE security environment and the security objectives.

585 For guidance on consistency analysis see Annex B.3.

AGD_ADM.1.8C

1:AGD_ADM.1-8 The evaluator ***shall examine*** the administrator guidance to determine that it describes all IT security requirements for the IT environment of the TOE that are relevant to the administrator.

586 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

587 This work unit relates to IT security requirements only and not to any organisational security policies.

588 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare them with the administrator guidance to ensure that all security requirements of the ST that are relevant to the administrator are described appropriately in the administrator guidance.

5.7.3 Evaluation of user guidance (AGD_USR.1)

5.7.3.1 Objectives

589 The objectives of this sub-activity are to determine whether the user guidance describes the security functions and interfaces provided by the TSF and whether this guidance provides instructions and guidelines for the secure use of the TOE.

5.7.3.2 Application notes

590 There may be different user roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF. The capabilities of these roles and their associated privileges are described in the FMT class. Different user roles and groups should be taken into consideration by the user guidance.

5.7.3.3 Input

591 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures.

5.7.3.4 Evaluator actions

592 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_USR.1.1E.

5.7.3.4.1 Action AGD_USR.1.1E

AGD_USR.1.1C

1:AGD_USR.1-1 The evaluator ***shall examine*** the user guidance to determine that it describes the security functions and interfaces available to the non-administrative users of the TOE.

593 The user guidance should contain an overview of the security functionality that is visible at the user interfaces.

594 The user guidance should identify and describe the purpose of the security interfaces and functions.

AGD_USR.1.2C

1:AGD_USR.1-2 The evaluator ***shall examine*** the user guidance to determine that it describes the use of user-accessible security functions provided by the TOE.

595 The user guidance should identify and describe the behaviour and interrelationship of the security interfaces and functions available to the user.

596 If the user is allowed to invoke a TOE security function, the user guidance provides a description of the interfaces available to the user for that function.

597 For each interface and function, the user guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system call, menu selection, command button);

b) describe the parameters to be set by the user and their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_USR.1.3C

1:AGD_USR.1-3 The evaluator ***shall examine*** the user guidance to determine that it contains warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

598 The configuration of the TOE may allow users to have dissimilar privileges in making use of the different functions of the TOE. This means that some users are authorised to perform certain functions, while other users may not be so authorised. These user-accessible functions and privileges are described by the user guidance.

599 The user guidance should identify the functions and privileges that can be used, the types of commands required for them, and the reasons for such commands. The user guidance should contain warnings regarding the use of the functions and privileges that must be controlled. Warnings should address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_USR.1.4C

1:AGD_USR.1-4 The evaluator ***shall examine*** the user guidance to determine that it presents all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

600 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the information that is of concern to the secure operation of the TOE need be included in the user guidance.

601 The user guidance should provide advice regarding effective use of the security functions (e.g. reviewing password composition practices, suggested frequency of user file backups, discussion on the effects of changing user access privileges).

602 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

603 The user guidance should indicate whether the user can invoke a function or whether the user requires the assistance of an administrator.

AGD_USR.1.5C

1:AGD_USR.1-5 The evaluator ***shall examine*** the user guidance to determine that it is consistent with all other documentation supplied for evaluation.

604 The evaluator ensures that the user guidance and all other documents supplied for evaluation do not contradict each other. This is especially true if the ST contains detailed information on any warnings to the TOE users with regard to the TOE security environment and the security objectives.

605 For guidance on consistency analysis see Annex B.3.

AGD_USR.1.6C

1:AGD_USR.1-6 The evaluator ***shall examine*** the user guidance to determine that it describes all security requirements for the IT environment of the TOE that are relevant to the user.

606 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

607 This work unit relates to IT security requirements only and not to any organisational security policies.

608 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare that with the user guidance to ensure that all security requirements of the ST, that are relevant to the user, are described appropriately in the user guidance.

5.8 Tests activity

609 The purpose of this activity is to determine, by independently testing a subset of the TSF, whether the TOE behaves as specified in the design documentation and in accordance with the TOE security functional requirements specified in the ST.

610 The tests activity at EAL1 contains a sub-activity related to the following component:

a) ATE_IND.1.

5.8.1 Application notes

611 The size and composition of the evaluator's test subset depends upon several factors discussed in the independent testing (ATE_IND.1) sub-activity. One such factor affecting the composition of the subset is *known public domain weaknesses,* information to which the evaluator needs access (e.g. from a scheme).

612 To create tests, the evaluator needs to understand the desired expected behaviour of a security function in the context of the requirements it is to satisfy. The evaluator may choose to focus on one security function of the TSF at a time, examining the ST requirement and the relevant parts of the functional specification and guidance documentation to gain an understanding of the way the TOE is expected to behave.

5.8.2 Evaluation of independent testing (ATE_IND.1)

5.8.2.1 Objectives

613 The objective of this sub-activity is to determine whether the TSF behaves as specified by independently testing a subset of the TSF.

5.8.2.2 Input

614 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures;

f) the TOE suitable for testing.

5.8.2.3 Evaluator actions

615 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ATE_IND.1.1E;

b) ATE_IND.1.2E.

5.8.2.3.1 Action ATE_IND.1.1E

ATE_IND.1.1C

1:ATE_IND.1-1 The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

616 The TOE used for evaluator testing should have the same unique reference as established by the ACM_CAP.1 sub-activity.

617 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator's TOE ~~verifies that there are~~ test configurations should be consistent with each evaluated configuration described in the ST.

618 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

619 If any test resources are used (e.g. meters, analysers) it will be the evaluator's responsibility to ensure that these resources are calibrated correctly.

1:ATE_IND.1-2 The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

620 It is possible for the evaluator to determine the state of the TOE in a number of ways. For example, previous successful completion of the ADO_IGS.1 sub-activity will satisfy this work unit if the evaluator still has confidence that the TOE being used for testing was installed properly and is in a known state. If this is not the case, then the evaluator should follow the developer's procedures to install, generate and start up the TOE, using the supplied guidance only.

621 If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed could satisfy work unit 1:ADO_IGS.1-2.

5.8.2.3.2 Action ATE_IND.1.2E

1:ATE_IND.1-3 The evaluator ***shall devise*** a test subset.

622 The evaluator selects a test subset and testing strategy that is appropriate for the TOE. One extreme testing strategy would be to have the test subset contain as many security functions as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few security functions based on their perceived relevance and rigorously test these functions.

623 Typically the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the security functional requirements identified in the ST using at least one test, but testing need not demonstrate exhaustive specification testing.

624 The evaluator, when selecting the subset of the TSF to be tested, should consider the following factors:

a) The number of security functions from which to draw upon for the test subset. Where the TOE includes only a small number of security functions, it may be practical to rigourously test all of the security functions. For TOEs with a large number of security functions this will not be cost-effective, and sampling is required.

b) Maintaining a balance of evaluation activities. Testing typically occupies 20-30% of the evaluator effort during the evaluation.

625 The evaluator selects the security functions to compose the subset. This selection will depend on a number of factors, and consideration of these factors may also influence the choice of test subset size:

a) Known public domain weaknesses commonly associated with the type of TOE (e.g. operating system, firewall). Know public domain weaknesses associated with the type of TOE will influence the selection process of the test subset. The evaluator should include those security functions that address known public domain weaknesses for that type of TOE in the subset (know public domain weaknesses in this context does not refer to vulnerabilities as such but to inadequacies or problem areas that have been experienced with this particular type of TOE). If no such weaknesses are known, then a more general approach of selecting a broad range of security functions may be more appropriate.

b) Significance of security functions. Those security functions more significant than others in terms of the security objectives for the TOE should be included in the test subset.

c) Complexity of the security function. Complex security functions may require complex tests that impose onerous requirements on the developer or evaluator, which will not be conducive to cost-effective evaluations. Conversely, complex security functions are a likely area to find errors and are good candidates for the subset. The evaluator will need to strike a balance between these considerations.

d) Implicit testing. Testing some security functions may often implicitly test other security functions, and their inclusion in the subset may maximize the number of security functions tested (albeit implicitly). Certain interfaces will typically be used to provide a variety of security functionality, and will tend to be the target of an effective testing approach.

e) Types of interfaces to the TOE (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

f) Functions that are innovative or unusual. Where the TOE contains innovative or unusual security functions, which may feature strongly in marketing literature, these should be strong candidates for testing.

626 This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

627 For guidance on sampling see Annex B.2.

1:ATE_IND.1-4 The evaluator ***shall produce*** test documentation for the test subset that is sufficiently detailed to enable the tests to be reproducible.

628 With an understanding of the expected behaviour of a security function, from the ST and the functional specification, the evaluator has to determine the most feasible way to test the function. Specifically the evaluator considers:

a) the approach that will be used, for instance, whether the security function will be tested at an external interface, at an internal interface using a test harness, or will an alternate test approach be employed (e.g. in exceptional circumstances, a code inspection);

b) the security function interface(s) that will be used to stimulate the security function and observe responses;

c) the initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

d) special test equipment that will be required to either stimulate a security function (e.g. packet generators) or make observations of a security function (e.g. network analysers).

629 The evaluator may find it practical to test each security function using a series of test cases, where each test case will test a very specific aspect of expected behaviour.

630 The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant design specification, and to the ST, if necessary.

1:ATE_IND.1-5 The evaluator ***shall conduct*** testing.

631 The evaluator uses the test documentation developed as a basis for executing tests on the TOE. The test documentation is used as a basis for testing but this does not preclude the evaluator from performing additional ad hoc tests. The evaluator may devise new tests based on behaviour of the TOE discovered during testing. These new tests are recorded in the test documentation.

1:ATE_IND.1-6 The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the security function behaviour to be tested;

b) instructions to connect and setup all required test equipment as required to conduct the test;

c) instructions to establish all prerequisite test conditions;

d) instructions to stimulate the security function;

e) instructions for observing the behaviour of the security function;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE;

h) actual test results.

632 The level of detail should be such that another evaluator could repeat the tests and obtain an equivalent result. While some specific details of the test results may be different (e.g. time and date fields in an audit record) the overall result should be identical.

633 There may be instances when it is unnecessary to provide all the information presented in this work unit (e.g. the actual test results of a test may not require any analysis before a comparison between the expected results can be made). The determination to omit this information is left to the evaluator, as is the justification.

1:ATE_IND.1-7 The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

634 Any differences in the actual and expected test results may indicate that the TOE does not perform as specified or that the evaluator test documentation may be incorrect. Unexpected actual results may require corrective maintenance to the

TOE or test documentation and perhaps require re-running of impacted tests and modifying the test sample size and composition. This determination is left to the evaluator, as is its justification.

1:ATE_IND.1-8 The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

635 The evaluator testing information reported in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing activity during the evaluation. The intent of providing this information is to give a meaningful overview of the testing effort. It is not intended that the information regarding testing in the ETR be an exact reproduction of specific test instructions or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the testing approach chosen, amount of testing performed, TOE test configurations, and the overall results of the testing activity.

636 Information that would typically be found in the ETR section regarding the evaluator testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested;

b) subset size chosen. The amount of security functions that were tested during the evaluation and a justification for the size;

c) selection criteria for the security functions that compose the subset. Brief statements about the factors considered when selecting security functions for inclusion in the subset;

d) security functions tested. A brief listing of the security functions that merited inclusion in the subset;

e) verdict for the activity. The overall judgement on the results of testing during the evaluation.

637 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the testing the evaluator performed during the evaluation.

Chapter 6

EAL2 evaluation

6.1 Introduction

638 EAL2 provides a low to moderate level of independently assured security. The security functions are analysed using a functional specification, guidance documentation, and the high-level design of the TOE to understand the security behaviour. The analysis is supported by independent testing of a subset of the TOE security functions, evidence of developer testing based on the functional specification, selective confirmation of the developer test results, analysis of strength of functions, and evidence of a developer search for obvious vulnerabilities. Further assurance is gained through a configuration list for the TOE and evidence of secure delivery procedures.

6.2 Objectives

639 The objective of this chapter is to define the minimal evaluation effort for achieving an EAL2 evaluation and to provide guidance on ways and means of accomplishing the evaluation.

6.3 EAL2 evaluation relationships

640 An EAL2 evaluation covers the following:

a) evaluation input task (Chapter 2);

b) EAL2 evaluation activities comprising the following:

  1) evaluation of the ST (Chapter 4);

  2) evaluation of the configuration management (Section 6.4);

  3) evaluation of the delivery and operation documents (Section 6.5);

  4) evaluation of the development documents (Section 6.6);

  5) evaluation of the guidance documents (Section 6.7);

  6) evaluation of the tests (Section 6.8);

  7) testing (Section 6.8);

  8) evaluation of the vulnerability assessment (Section 6.9);

c) evaluation output task (Chapter 2).

641 The evaluation activities are derived from the EAL2 assurance requirements contained in the CC Part 3.

642 The ST evaluation is started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities.

643 The sub-activities comprising an EAL2 evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator.

644 For guidance on dependencies see Annex B.4.

6.4 Configuration management activity

645 The purpose of the configuration management activity is to assist the consumer in identifying the evaluated TOE, and to ensure that configuration items are uniquely identified.

646 The configuration management activity at EAL2 contains a sub-activity related to the following component:

a) ACM_CAP.2.

6.4.1 Evaluation of CM capabilities (ACM_CAP.2)

6.4.1.1 Objectives

647 The objectives of this sub-activity are to determine whether the developer has clearly identified the TOE and its associated configuration items.

6.4.1.2 Application notes

648 This component contains an implicit evaluator action to determine that the CM system is being used. As the requirements here are limited to identification of the TOE and provision of a configuration list, this action is already covered by, and limited to, the existing work units. At ACM_CAP.3 the requirements are expanded beyond these two items, and more explicit evidence of operation is required.

6.4.1.3 Input

649 The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

6.4.1.4 Evaluator actions

650 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_CAP.2.1E.

6.4.1.4.1 Action ACM_CAP.2.1E

ACM_CAP.2.1C

2:ACM_CAP.2-1 The evaluator ***shall check*** that the version of the TOE provided for evaluation is uniquely referenced.

651 The evaluator should use the developer's CM system to validate the uniqueness of the reference by checking the configuration list to ensure that the configuration items are uniquely identified. Evidence that the version provided for evaluation is uniquely referenced may be incomplete if only one version is examined during the evaluation, and the evaluator should look for a referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates). However, the absence of any reference will normally lead to a fail verdict against this requirement unless the evaluator is confident that the TOE can be uniquely identified.

652 The evaluator should seek to examine more than one version of the TOE (e.g. during rework following discovery of a vulnerability), to check that the two versions are referenced differently.

653 ACM_CAP.2.2C

2:ACM_CAP.2-2 The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

654 The evaluator should ensure that the TOE contains a unique reference such that it is possible to distinguish different versions of the TOE. This could be achieved through labelled packaging or media, or by a label displayed by the operational TOE. This is to ensure that it would be possible for consumers to identify the TOE (e.g. at the point of purchase or use).

655 The TOE may provide a method by which it can be easily identified. For example, a software TOE may display its name and version number during the start up routine, or in response to a command line entry. A hardware or firmware TOE may be identified by a part number physically stamped on the TOE.

2:ACM_CAP.2-3 The evaluator ***shall check*** that the TOE references used are consistent.

656 If the TOE is labelled more than once then the labels have to be consistent. For example, it should be possible to relate any labelled guidance documentation supplied as part of the TOE to the evaluated operational TOE. This ensures that consumers can be confident that they have purchased the evaluated version of the TOE, that they have installed this version, and that they have the correct version of the guidance to operate the TOE in accordance with its ST. The evaluator can use the configuration list that is part of the provided CM documentation to verify the consistent use of identifiers.

657 The evaluator also verifies that the TOE reference is consistent with the ST.

658 For guidance on consistency analysis see Annex B.3.

ACM_CAP.2.3C

2:ACM_CAP.2-4 The evaluator ***shall check*** that the CM documentation provided includes a configuration list.

659 A configuration list identifies the items being maintained under configuration control.

Interp Note : The following element is added as a result of Interpretation 003.

ACM_CAP.2.<new C&P element resulting from Interp>C

2:ACM_CAP.2-new The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.

The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.

ACM_CAP.2.4C

2:ACM_CAP.2-5 The evaluator ***shall examine*** the configuration list to determine that it identifies the configuration items that comprise the TOE.

660 The minimum scope of configuration items to be covered in the configuration list is given by ACM_SCP. If no ACM_SCP component is included, the evaluator should assess the adequacy of the list on the basis of the approach taken by the developer to CM, taking the requirements of ACM_SCP.1 as an upper bound (since it would be unreasonable to expect more than is required there). For example, when a change is made to the TOE or any item of documentation, the evaluator may observe or enquire at what level of granularity the item is re-issued. This granularity should correspond to the configuration items that appear in the configuration list.

ACM_CAP.2.5C

2:ACM_CAP.2-6 The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

ACM_CAP.2.6C

Interp Note : The following element is replaced as a result of Interpretation 003.

2:ACM_CAP.2-7 ~~The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.~~ The evaluator shall examine the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

661 ~~The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.~~Assurance that the CM system

uniquely identifies all configuration items is gained by examining the identifiers for the configuration items. For both configuration items that comprise the TOE, and drafts of configuration items that are submitted by the developer as evaluation evidence, the evaluator confirms that each configuration item possesses a unique identifier in a manner consistent with the unique identification method that is described in the CM documentation.

6.5 Delivery and operation activity

662 The purpose of the delivery and operation activity is to judge the adequacy of the documentation of the procedures used to ensure that the TOE is installed, generated, and started in the same way the developer intended it to be and that it is delivered without modification. This includes both the procedures taken while the TOE is in transit, as well as the initialisation, generation, and start-up procedures.

663 The delivery and operation activity at EAL2 contains sub-activities related to the following components:

a) ADO_DEL.1;

b) ADO_IGS.1.

6.5.1 Evaluation of delivery (ADO_DEL.1)

6.5.1.1 Objectives

Interp Note : The following paragraph is changed as a result of Interpretation 016.

664 The objective of this sub-activity is to determine whether the delivery documentation describes all procedures used to maintain ~~integrity~~ security of the TOE when distributing the TOE to the user's site.

6.5.1.2 Input

665 The evaluation evidence for this sub-activity is:

a) the delivery documentation.

6.5.1.3 Evaluator actions

666 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_DEL.1.1E;

b) implied evaluator action based on ADO_DEL.1.2D.

6.5.1.3.1 Action ADO_DEL.1.1E

ADO_DEL.1.1C

2:ADO_DEL.1-1 The evaluator ***shall examine*** the delivery documentation to determine that it describes all procedures that are necessary to maintain security when distributing versions of the TOE or parts of it to the user's site.

667 Interpretation of the term *necessary* will need to consider the nature of the TOE and information contained in the ST. The level of protection provided should be commensurate with the assumptions, threats, organisational security policies, and

security objectives identified in the ST. In some cases these may not be explicitly expressed in relation to delivery. The evaluator should determine that a balanced approach has been taken, such that delivery does not present an obvious weak point in an otherwise secure development process.

Interp Note : The following two paragraphs are changed as a result of Interpretation 016.

668 The delivery procedures describe proper procedures to determine the identification of the TOE and to maintain ~~integrity~~ security of the TOE during transfer of the TOE or its component parts. The procedures describe which parts of the TOE need to be covered by these procedures. It should contain procedures for physical or electronic (e.g. for downloading off the Internet) distribution where applicable. The delivery procedures refer to the entire TOE, including applicable software, hardware, firmware and documentation.

669 ~~The emphasis on integrity is not surprising, since integrity will always be of concern for TOE delivery. Where confidentiality and availability are of concern, they also should be considered under this work unit.~~ The emphasis in the delivery documentation is likely to be on measures related to integrity, as technical measures are required to be applied to maintain integrity during the TOE delivery. However, confidentiality and availability of the delivery will be of concern in the delivery of some TOEs; procedures relating to these aspects of the secure delivery should also be discussed in the procedures.

670 The delivery procedures should be applicable across all phases of delivery from the production environment to the installation environment (e.g. packaging, storage and distribution).

Interp Note : The following two paragraphs are added as a result of Interpretation 116.

Standard commercial practice for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution, the public mail or a private distribution service may be acceptable.

The suitability of the choice of the delivery procedures is influenced by the TOE (e.g. whether it is software or hardware) and by the security objectives. In cases where the delivery procedures differ for different parts of the TOE, the totality of procedures are suitable to meet the overall security objectives.

Interp Note : The following work unit and its two guidance are deleted as a result of Interpretation 116.

2:ADO_DEL.1-2 ~~The evaluator ***shall examine*** the delivery procedures to determine that the chosen procedure and the part of the TOE it covers is suitable to meet the security objectives.~~

671 ~~The suitability of the choice of the delivery procedures is influenced by the specific TOE (e.g. whether it is software or hardware) and by the security objectives.~~

672 ~~Standard commercial practice for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution the public mail or a private distribution service may be acceptable.~~

6.5.1.3.2 Implied evaluator action

ADO_DEL.1.2D

2:ADO_DEL.1-3 The evaluator ***shall examine*** aspects of the delivery process to determine that the delivery procedures are used.

673 The approach taken by the evaluator to check the application of delivery procedures will depend on the nature of the TOE, and the delivery process itself. In addition to examination of the procedures themselves, the evaluator should seek some assurance that they are applied in practice. Some possible approaches are:

a) a visit to the distribution site(s) where practical application of the procedures may be observed;

b) examination of the TOE at some stage during delivery, or at the user's site (e.g. checking for tamper proof seals);

c) observing that the process is applied in practice when the evaluator obtains the TOE through regular channels;

d) questioning end users as to how the TOE was delivered.

674 For guidance on site visits see Annex B.5.

675 It may be the case of a newly developed TOE that the delivery procedures have yet to be exercised. In these cases, the evaluator has to be satisfied that appropriate procedures and facilities are in place for future deliveries and that all personnel involved are aware of their responsibilities. The evaluator may request a "dry run" of a delivery if this is practical. If the developer has produced other similar products, then an examination of procedures in their use may be useful in providing assurance.

6.5.2 Evaluation of installation, generation and start-up (ADO_IGS.1)

6.5.2.1 Objectives

676 The objective of this sub-activity is to determine whether the procedures and steps for the secure installation, generation, and start-up of the TOE have been documented and result in a secure configuration.

6.5.2.2 Input

677 The evaluation evidence for this sub-activity is:

a) the administrator guidance;

b) the secure installation, generation, and start-up procedures;

c) the TOE suitable for testing.

6.5.2.3 Application notes

678 The installation, generation, and start-up procedures refer to all installation, generation, and start-up procedures, regardless of whether they are performed at the user's site or at the development site that are necessary to progress the TOE to the secure configuration as described in the ST.

6.5.2.4 Evaluator actions

679 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_IGS.1.1E;

b) ADO_IGS.1.2E.

6.5.2.4.1 Action ADO_IGS.1.1E

ADO_IGS.1.1C

2:ADO_IGS.1-1 The evaluator ***shall check*** that the procedures necessary for the secure installation, generation and start-up of the TOE have been provided.

680 If it is not anticipated that the installation, generation, and start-up procedures will or can be re-applied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

6.5.2.4.2 Action ADO_IGS.1.2E

2:ADO_IGS.1-2 The evaluator ***shall examine*** the provided installation, generation, and start-up procedures to determine that they describe the steps necessary for secure installation, generation, and start-up of the TOE.

681 If it is not anticipated that the installation, generation, and start-up procedures will or can be re-applied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

682 The installation, generation, and start-up procedures may provide detailed information about the following:

a) changing the installation specific security characteristics of entities under the control of the TSF;

b) handling exceptions and problems;

c) minimum system requirements for secure installation if applicable.

683 In order to confirm that the installation, generation, and start-up procedures result in a secure configuration, the evaluator may follow the developer's procedures and may perform the activities that customers are usually expected to perform to install, generate, and start-up the TOE (if applicable to the TOE), using the supplied guidance documentation only. This work unit might be performed in conjunction with the 2:ATE_IND.2-2 work unit.

6.6 Development activity

684 The purpose of the development activity is to assess the design documentation in terms of its adequacy to understand how the TSF provides the security functions of the TOE. This understanding is achieved through examination of increasingly refined descriptions of the TSF design documentation. Design documentation consists of a functional specification (which describes the external interfaces of the TOE) and a high-level design (which describes the architecture of the TOE in terms of internal subsystems). There is also a representation correspondence (which maps representations of the TOE to one another in order to ensure consistency).

685 The development activity at EAL2 contains sub-activities related to the following components:

a) ADV_FSP.1;

b) ADV_HLD.1;

c) ADV_RCR.1.

6.6.1 Application notes

686 The CC requirements for design documentation are levelled by formality. The CC considers a document's degree of formality (that is, whether it is informal, semiformal or formal) to be hierarchical. An informal document is one that is expressed in a natural language. The methodology does not dictate the specific language that must be used; that issue is left for the scheme. The following paragraphs differentiate the contents of the different informal documents.

687 An informal functional specification comprises a description the security functions (at a level similar to that of the TOE summary specification) and a description of the externally-visible interfaces to the TSF. For example, if an operating system presents the user with a means of self-identification, of creating files, of modifying or deleting files, of setting permissions defining what other users may access files, and of communicating with remote machines, its functional specification would contain descriptions of each of these functions. If there are also audit functions that detect and record the occurrences of such events, descriptions of these audit functions would also be expected to be part of the functional specification; while these functions are technically not directly invoked by the user at the external interface, they certainly are affected by what occurs at the user's external interface.

688 An informal high-level design is expressed in terms of sequences of actions that occur in each subsystem in response to stimulus at its interface. For example, a firewall might be composed of subsystems that deal with packet filtering, with remote administration, with auditing, and with connection-level filtering. The high-level design description of the firewall would describe the actions that are taken, in terms of what actions each subsystem takes when an incoming packet arrives at the firewall.

689 Informality of the demonstration of correspondence need not be in a prose form; a simple two-dimensional mapping may be sufficient. For example, a matrix with modules listed along one axis and subsystems listed along the other, with the cells identifying the correspondence of the two, would serve to provide an adequate informal correspondence between the high-level design and the low-level design.

6.6.2 Evaluation of functional specification (ADV_FSP.1)

6.6.2.1 Objectives

690 The objective of this sub-activity is to determine whether the developer has provided an adequate description of the security functions of the TOE and whether the security functions provided by the TOE are sufficient to satisfy the security functional requirements of the ST.

6.6.2.2 Input

691 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance.

6.6.2.3 Evaluator actions

692 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_FSP.1.1E;

b) ADV_FSP.1.2E.

6.6.2.3.1 Action ADV_FSP.1.1E

ADV_FSP.1.1C

2:ADV_FSP.1-1 The evaluator ***shall examine*** the functional specification to determine that it contains all necessary informal explanatory text.

693 If the entire functional specification is informal, this work unit is not applicable and is therefore considered to be satisfied.

694 Supporting narrative descriptions are necessary for those portions of the functional specification that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_FSP.1.2C

2:ADV_FSP.1-2 The evaluator ***shall examine*** the functional specification to determine that it is internally consistent.

695 The evaluator validates the functional specification by ensuring that the descriptions of the interfaces making up the TSFI are consistent with the descriptions of the functions of the TSF.

696 For guidance on consistency analysis see Annex B.3.

ADV_FSP.1.3C

2:ADV_FSP.1-3 The evaluator ***shall examine*** the functional specification to determine that it identifies all of the external TOE security function interfaces.

697 The term *external* refers to that which is visible to the user. External interfaces to the TOE are either direct interfaces to the TSF or interfaces to non-TSF portions of the TOE. However, these non-TSF interfaces might have eventual access to the TSF. These external interfaces that directly or indirectly access the TSF collectively make up the TOE security function interface (TSFI). Figure 6.1 shows a TOE with TSF (shaded) portions and non-TSF (empty) portions. This TOE has three external interfaces: interface *c* is a direct interface to the TSF; interface *b* is an indirect interface to the TSF; and interface *a* is an interface to non-TSF portions of the TOE. Therefore, interfaces *b* and *c* make up the TFSI.

698 It should be noted that all security functions reflected in the functional requirements of CC Part 2 (or in extended components thereof) will have some sort of externally-visible manifestation. While not all of these are necessarily interfaces from which the security function can be tested, they are all externally-visible to some extent and must therefore be included in the functional specification.

699 For guidance on determining the TOE boundary see Annex B.6.

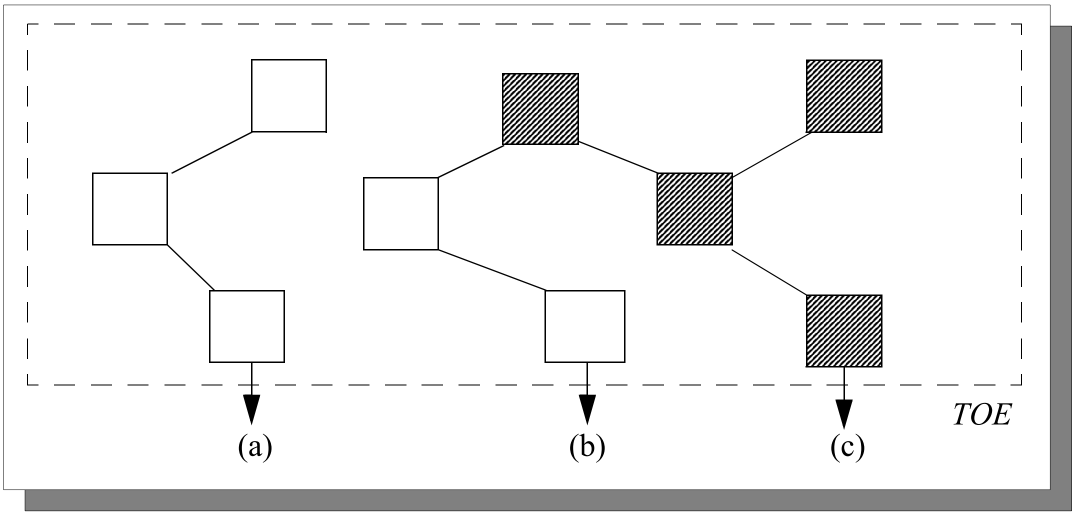

Figure 6.1 TSF Interfaces

1:ADV_FSP.1-4 The evaluator ***shall examine*** the functional specification to determine that it describes all of the external TOE security function interfaces.

700 For a TOE that has no threat of malicious users (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are rightfully excluded from its ST), the only interfaces that are described in the functional specification (and expanded upon in the other TSF representation descriptions) are those to and from the TSF. The absence of FPT_PHP, FPT_RVM, and FPT_SEP presumes there is no concern for any sort of bypassing of the security features; therefore, there is no concern with any possible impact that other interfaces might have on the TSF.

701 On the other hand, if the TOE has a threat of malicious users or bypass (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are included in its ST), all external interfaces are described in the functional specification, but only to the extent that the effect of each is made clear: interfaces to the security functions (i.e. interfaces *b* and *c* in Figure 6.1) are completely described, while other interfaces are described only to the extent that it is clear that the TSF is inaccessible through the interface (i.e. that the interface is of type *a*, rather than *b* in Figure 6.1). The inclusion of FPT_PHP, FPT_RVM, and FPT_SEP implies a concern that all interfaces might have some effect upon the TSF. Because each external interface is a potential TSF interface, the functional specification must contain a description of each interface in sufficient detail so that an evaluator can determine whether the interface is security relevant.

702 Some architectures lend themselves to readily provide this interface description in sufficient detail for groups of external interfaces. For example, a kernel architecture is such that all calls to the operating system are handled by kernel

programs; any calls that might violate the TSP must be called by a program with the privilege to do so. All programs that execute with privilege must be included in the functional specification. Any program external to the kernel that executes without privilege is incapable of affecting the TSP (i.e. such programs are interfaces of type *a*, rather than *b* in Figure 6.1) and may, therefore, be excluded from the functional specification. It is worth noting that, while the evaluator's understanding of the interface description can be expedited in cases where there is a kernel architecture, such an architecture is not necessary.

2:ADV_FSP.1-5 The evaluator ***shall examine*** the presentation of the TSFI to determine that it adequately and correctly describes the behaviour of the TOE at each external interface describing effects, exceptions and error messages.

703 In order to assess the adequacy and correctness of an interface's presentation, the evaluator uses the functional specification, the TOE summary specification of the ST, and the user and administrator guidance to assess the following factors:

a) All security relevant user input parameters (or a characterisation of those parameters) should be identified. For completeness, parameters outside of direct user control should be identified if they are usable by administrators.

b) All security relevant behaviour described in the reviewed guidance should be reflected in the description of semantics in the functional specification. This should include an identification of the behaviour in terms of events and the effect of each event. For example, if an operating system provides a rich file system interface, where it provides a different error code for each reason why a file is not opened upon request (e.g. access denied, no such file, file is in use by another user, user is not authorised to open the file after 5pm, etc.), the functional specification should explain that a file is either opened upon request, or else that an error code is returned. (While the functional specification may enumerate all these different reasons for errors, it need not provide such detail.) The description of the semantics should include how the security requirements apply to the interface (e.g. whether the use of the interface is an auditable event and, if so, the information that can be recorded).

c) All interfaces are described for all possible modes of operation. If the TSF provides the notion of privilege, the description of the interface should explain how the interface behaves in the presence or absence of privilege.

d) The information contained in the descriptions of the security relevant parameters and syntax of the interface should be consistent across all documentation.

704 Verification of the above is done by reviewing the functional specification and the TOE summary specification of the ST, as well as the user and administrator guidance provided by the developer. For example, if the TOE were an operating system and its underlying hardware, the evaluator would look for discussions of user-accessible programs, descriptions of protocols used to direct the activities of programs, descriptions of user-accessible databases used to direct the activities of

programs, and for user interfaces (e.g. commands, application program interfaces) as applicable to the TOE under evaluation; the evaluator would also ensure that the processor instruction set is described.

705 This review might be iterative, such that the evaluator would not discover the functional specification to be incomplete until the design, source code, or other evidence is examined and found to contain parameters or error messages that have been omitted from the functional specification.

ADV_FSP.1.4C

2:ADV_FSP.1-6 The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

706 In order to assess the completeness of the TSF representation, the evaluator consults the TOE summary specification of the ST, the user guidance, and the administrator guidance. None of these should describe security functions that are absent from the TSF presentation of the functional specification.

6.6.2.3.2 Action ADV_FSP.1.2E

2:ADV_FSP.1-7 The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the TOE security functional requirements.

707 To ensure that all ST security functional requirements are covered by the functional specification, the evaluator may construct a map between the TOE summary specification and the functional specification. Such a map might be already provided by the developer as evidence for meeting the correspondence (ADV_RCR.*) requirements, in which case the evaluator need only verify the completeness of this mapping, ensuring that all security functional requirements are mapped onto applicable TSFI presentations in the functional specification.

2:ADV_FSP.1-8 The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the TOE security functional requirements.

708 For each interface to a security function with specific characteristics, the detailed information in the functional specification must be exactly as it is specified in the ST. For example, if the ST contains user authentication requirements that the password length must be eight characters, the TOE must have eight-character passwords; if the functional specification describes six-character fixed length passwords, the functional specification would not be an accurate instantiation of the requirements.

709 For each interface in the functional specification that operates on a controlled resource, the evaluator determines whether it returns an error code that indicates a possible failure due to enforcement of one of the security requirements; if no error code is returned, the evaluator determines whether an error code should be returned. For example, an operating system might present an interface to OPEN a controlled object. The description of this interface may include an error code that indicates that access was not authorised to the object. If such an error code does

not exist, the evaluator should confirm that this is appropriate (because, perhaps, access mediation is performed on READs and WRITEs, rather than on OPENs).

6.6.3 Evaluation of high-level design (ADV_HLD.1)

6.6.3.1 Objectives

710 The objective of this sub-activity is to determine whether the high-level design provides a description of the TSF in terms of major structural units (i.e. subsystems), and is a correct realisation of the functional specification.

6.6.3.2 Input

711 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design.

6.6.3.3 Evaluator actions

712 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_HLD.1.1E;

b) ADV_HLD.1.2E.

6.6.3.3.1 Action ADV_HLD.1.1E

ADV_HLD.1.1C

2:ADV_HLD.1-1 The evaluator ***shall examine*** the high-level design to determine that it contains all necessary informal explanatory text.

713 If the entire high-level design is informal, this work unit is not applicable and is therefore considered to be satisfied.

714 Supporting narrative descriptions are necessary for those portions of the high-level design that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_HLD.1.2C

2:ADV_HLD.1-2 The evaluator ***shall examine*** the presentation of the high-level design to determine that it is internally consistent.

715 For guidance on consistency analysis see Annex B.3.

716 The evaluator validates the subsystem interface specifications by ensuring that the interface specifications are consistent with the description of the purpose of the subsystem.

ADV_HLD.1.3C

2:ADV_HLD.1-3 The evaluator ***shall examine*** the high-level design to determine that the TSF is described in terms of subsystems.

717 With respect to the high-level design, the term *subsystem* refers to large, related units (such as memory-management, file-management, process-management). Breaking a design into the basic functional areas aids in the understanding of the design.

718 The primary purpose for examining the high-level design is to aid the evaluator's understanding of the TOE. The developer's choice of subsystem definition, and of the grouping of TSFs within each subsystem, are an important aspect of making the high-level design useful in understanding the TOE's intended operation. As part of this work unit, the evaluator should make an assessment as to the appropriateness of the number of subsystems presented by the developer, and also of the choice of grouping of functions within subsystems. The evaluator should ensure that the decomposition of the TSF into subsystems is sufficient for the evaluator to gain a high-level understanding of how the functionality of the TSF is provided.

719 The subsystems used to describe the high-level design need not be called "subsystems", but should represent a similar level of decomposition. For example, the design may be decomposed using "layers" or "managers".

ADV_HLD.1.4C

2:ADV_HLD.1-4 The evaluator ***shall examine*** the high-level design to determine that it describes the security functionality of each subsystem.

720 The security functional behaviour of a subsystem is a description of what the subsystem does. This should include a description of any actions that the subsystem may be directed to perform through its functions and the effects the subsystem may have on the security state of the TOE (e.g. changes in subjects, objects, security databases).

ADV_HLD.1.5C

2:ADV_HLD.1-5 The evaluator ***shall check*** the high-level design to determine that it identifies all hardware, firmware, and software required by the TSF.

721 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

722 If the ST contains the optional statement of security requirements for the IT environment, the evaluator compares the list of hardware, firmware, or software required by the TSF as stated in the high-level design to the statement of security requirements for the IT environment to determine that they agree. The information in the ST characterises the underlying abstract machine on which the TOE will execute.

723 If the high-level design includes security requirements for the IT environment that are not included in the ST, or if they differ from those included in the ST, this inconsistency is assessed by the evaluator under Action ADV_HLD.1.2E.

2:ADV_HLD.1-6 The evaluator ***shall examine*** the high-level design to determine that it includes a presentation of the functions provided by the supporting protection mechanisms implemented in the underlying hardware, firmware, or software.

724 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

725 The presentation of the functions provided by the underlying abstract machine on which the TOE executes need not be at the same level of detail as the presentation of functions that are part of the TSF. The presentation should explain how the TOE uses the functions provided in the hardware, firmware, or software that implement the security requirements for the IT environment that the TOE is dependent upon to support the TOE security objectives.

726 The statement of security requirements for the IT environment may be abstract, particularly if it is intended to be capable of being satisfied by a variety of different combinations of hardware, firmware, or software. As part of the Tests activity, where the evaluator is provided with at least one instance of an underlying machine that is claimed to satisfy the security requirements for the IT environment, the evaluator can determine whether it provides the necessary security functions for the TOE. This determination by the evaluator does not require testing or analysis of the underlying machine; it is only a determination that the functions expected to be provided by it actually exist.

ADV_HLD.1.6C

2:ADV_HLD.1-7 The evaluator ***shall check*** that the high-level design identifies the interfaces to the TSF subsystems.

727 The high-level design includes, for each subsystem, the name of each of its entry points.

ADV_HLD.1.7C

2:ADV_HLD.1-8 The evaluator ***shall check*** that the high-level design identifies which of the interfaces to the subsystems of the TSF are externally visible.

6.6.3.3.2 Action ADV_HLD.1.2E

2:ADV_HLD.1-9 The evaluator ***shall examine*** the high-level design to determine that it is an accurate instantiation of the TOE security functional requirements.

728 The evaluator analyses the high-level design for each TOE security function to ensure that the function is accurately described. The evaluator also ensures that the function has no dependencies that are not included in the high-level design.

729 The evaluator also analyses the security requirements for the IT environment in both the ST and the high-level design to ensure that they agree. For example, if the ST includes TOE security functional requirements for the storage of an audit trail, and the high-level design stated that audit trail storage is provided by the IT environment, then the high-level design is not an accurate instantiation of the TOE security functional requirements.

2:ADV_HLD.1-10 The evaluator ***shall examine*** the high-level design to determine that it is a complete instantiation of the TOE security functional requirements.

730 To ensure that all ST security functional requirements are covered by the high-level design, the evaluator may construct a map between the TOE security functional requirements and the high-level design.

6.6.4 Evaluation of representation correspondence (ADV_RCR.1)

6.6.4.1 Objectives

731 The objective of this sub-activity is to determine whether the developer has correctly and completely implemented the requirements of the ST and functional specification in the high-level design.

6.6.4.2 Input

732 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the correspondence analysis between the TOE summary specification and the functional specification;

e) the correspondence analysis between the functional specification and the high-level design.

6.6.4.3 Evaluator actions

733 This sub-activity comprises one CC Part 3 evaluator action element:

a) ADV_RCR.1.1E.

6.6.4.3.1 Action ADV_RCR.1.1E

ADV_RCR.1.1C

2:ADV_RCR.1-1 The evaluator ***shall examine*** the correspondence analysis between the TOE summary specification and the functional specification to determine that the functional specification is a correct and complete representation of the TOE security functions.

734 The evaluator's goal in this work unit is to determine that all security functions identified in the TOE summary specification are represented in the functional specification and that they are represented accurately.

735 The evaluator reviews the correspondence between the TOE security functions of the TOE summary specification and the functional specification. The evaluator looks for consistency and accuracy in the correspondence. Where the correspondence analysis indicates a relationship between a security function of the TOE summary specification and an interface description in the functional specification, the evaluator verifies that the security functionality of both are the

same. If the security functions of the TOE summary specification are correctly and completely present in the corresponding interface, this work unit will be satisfied.

736 This work unit may be done in conjunction with work units 2:ADV_FSP.1-7 and 2:ADV_FSP.1-8.

2:ADV_RCR.1-2 The evaluator ***shall examine*** the correspondence analysis between the functional specification and the high-level design to determine that the high-level design is a correct and complete representation of the functional specification.

737 The evaluator uses the correspondence analysis, the functional specification, and the high-level design to ensure that it is possible to map each security function identified in the functional specification onto a TSF subsystem described in the high-level design. For each security function, the correspondence indicates which TSF subsystems are involved in the support of the function. The evaluator verifies that the high-level design includes a description of a correct realisation of each security function.

6.7 Guidance documents activity

738 The purpose of the guidance document activity is to judge the adequacy of the documentation describing how to use the operational TOE. Such documentation includes both that aimed at trusted administrators and non-administrator users whose incorrect actions could adversely affect the security of the TOE, as well as that aimed at untrusted users whose incorrect actions could adversely affect the security of their own data.

739 The guidance documents activity at EAL2 contains sub-activities related to the following components:

a) AGD_ADM.1;

b) AGD_USR.1.

6.7.1 Application notes

740 The guidance documents activity applies to those functions and interfaces which are related to the security of the TOE. The secure configuration of the TOE is described in the ST.

6.7.2 Evaluation of administrator guidance (AGD_ADM.1)

6.7.2.1 Objectives

741 The objective of this sub-activity is to determine whether the administrator guidance describes how to administer the TOE in a secure manner.

6.7.2.2 Application notes

742 The term *administrator* is used to indicate a human user who is trusted to perform security critical operations within the TOE, such as setting TOE configuration parameters. The operations may affect the enforcement of the TSP, and the administrator therefore possesses specific privileges necessary to perform those operations. The role of the administrator(s) has to be clearly distinguished from the role of non-administrative users of the TOE.

743 There may be different administrator roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF such as auditor, administrator, or daily-management. Each role can encompass an extensive set of capabilities, or can be a single one. The capabilities of these roles and their associated privileges are described in the FMT class. Different administrator roles and groups should be taken into consideration by the administrator guidance.

6.7.2.3 Input

744 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures.

6.7.2.4 Evaluator actions

745 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_ADM.1.1E.

6.7.2.4.1 Action AGD_ADM.1.1E

AGD_ADM.1.1C

2:AGD_ADM.1-1 The evaluator ***shall examine*** the administrator guidance to determine that it describes the administrative security functions and interfaces available to the administrator of the TOE.

746 The administrator guidance should contain an overview of the security functionality that is visible at the administrator interfaces.

747 The administrator guidance should identify and describe the purpose, behaviour, and interrelationships of the administrator security interfaces and functions.

748 For each administrator security interface and function, the administrator guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system calls, menu selection, command button);

b) describe the parameters to be set by the administrator, their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_ADM.1.2C

2:AGD_ADM.1-2 The evaluator ***shall examine*** the administrator guidance to determine that it describes how to administer the TOE in a secure manner.

749 The administrator guidance describes how to operate the TOE according to the TSP in an IT environment that is consistent with the one described in the ST.

AGD_ADM.1.3C

2:AGD_ADM.1-3 The evaluator ***shall examine*** the administrator guidance to determine that it contains warnings about functions and privileges that should be controlled in a secure processing environment.

750 The configuration of the TOE may allow users to have dissimilar privileges to make use of the different functions of the TOE. This means that some users may be authorised to perform certain functions while other users may not be so authorised. These functions and privileges should be described by the administrator guidance.

751 The administrator guidance identifies the functions and privileges that must be controlled, the types of controls required for them, and the reasons for such controls. Warnings address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_ADM.1.4C

2:AGD_ADM.1-4 The evaluator ***shall examine*** the administrator guidance to determine that it describes all assumptions regarding user behaviour that are relevant to the secure operation of the TOE.

752 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the information that is of concern to the secure operation of the TOE need be included in the administrator guidance.

753 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

AGD_ADM.1.5C

2:AGD_ADM.1-5 The evaluator ***shall examine*** the administrator guidance to determine that it describes all security parameters under the control of the administrator indicating secure values as appropriate.

754 For each security parameter, the administrator guidance should describe the purpose of the parameter, the valid and default values of the parameter, and secure and insecure use settings of such parameters, both individually or in combination.

AGD_ADM.1.6C

2:AGD_ADM.1-6 The evaluator ***shall examine*** the administrator guidance to determine that it describes each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

755 All types of security-relevant events are detailed, such that an administrator knows what events may occur and what action (if any) the administrator may have to take in order to maintain security. Security-relevant events that may occur during

operation of the TOE (e.g. audit trail overflow, system crash, updates to user records, such as when a user account is removed when the user leaves the organisation) are adequately defined to allow administrator intervention to maintain secure operation.

AGD_ADM.1.7C

2:AGD_ADM.1-7 The evaluator ***shall examine*** the administrator guidance to determine that it is consistent with all other documents supplied for evaluation.

756 The ST in particular may contain detailed information on any warnings to the TOE administrators with regard to the TOE security environment and the security objectives.

757 For guidance on consistency analysis see Annex B.3.

AGD_ADM.1.8C

2:AGD_ADM.1-8 The evaluator ***shall examine*** the administrator guidance to determine that it describes all IT security requirements for the IT environment of the TOE that are relevant to the administrator.

758 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

759 This work unit relates to IT security requirements only and not to any organisational security policies.

760 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare them with the administrator guidance to ensure that all security requirements of the ST that are relevant to the administrator are described appropriately in the administrator guidance.

6.7.3 Evaluation of user guidance (AGD_USR.1)

6.7.3.1 Objectives

761 The objectives of this sub-activity are to determine whether the user guidance describes the security functions and interfaces provided by the TSF and whether this guidance provides instructions and guidelines for the secure use of the TOE.

6.7.3.2 Application notes

762 There may be different user roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF. The capabilities of these roles and their associated privileges are described in the FMT class. Different user roles and groups should be taken into consideration by the user guidance.

6.7.3.3 Input

763 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures.

6.7.3.4 Evaluator actions

764 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_USR.1.1E.

6.7.3.4.1 Action AGD_USR.1.1E

AGD_USR.1.1C

2:AGD_USR.1-1 The evaluator ***shall examine*** the user guidance to determine that it describes the security functions and interfaces available to the non-administrative users of the TOE.

765 The user guidance should contain an overview of the security functionality that is visible at the user interfaces.

766 The user guidance should identify and describe the purpose of the security interfaces and functions.

AGD_USR.1.2C

2:AGD_USR.1-2 The evaluator ***shall examine*** the user guidance to determine that it describes the use of user-accessible security functions provided by the TOE.

767 The user guidance should identify and describe the behaviour and interrelationship of the security interfaces and functions available to the user.

768 If the user is allowed to invoke a TOE security function, the user guidance provides a description of the interfaces available to the user for that function.

769 For each interface and function, the user guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system call, menu selection, command button);

b) describe the parameters to be set by the user and their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_USR.1.3C

2:AGD_USR.1-3 The evaluator ***shall examine*** the user guidance to determine that it contains warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

770 The configuration of the TOE may allow users to have dissimilar privileges in making use of the different functions of the TOE. This means that some users are authorised to perform certain functions, while other users may not be so authorised. These user-accessible functions and privileges are described by the user guidance.

771 The user guidance should identify the functions and privileges that can be used, the types of commands required for them, and the reasons for such commands. The user guidance should contain warnings regarding the use of the functions and privileges that must be controlled. Warnings should address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_USR.1.4C

2:AGD_USR.1-4 The evaluator ***shall examine*** the user guidance to determine that it presents all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

772 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the

information that is of concern to the secure operation of the TOE need be included in the user guidance.

773 The user guidance should provide advice regarding effective use of the security functions (e.g. reviewing password composition practices, suggested frequency of user file backups, discussion on the effects of changing user access privileges).

774 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

775 The user guidance should indicate whether the user can invoke a function or whether the user requires the assistance of an administrator.

AGD_USR.1.5C

2:AGD_USR.1-5 The evaluator ***shall examine*** the user guidance to determine that it is consistent with all other documentation supplied for evaluation.

776 The evaluator ensures that the user guidance and all other documents supplied for evaluation do not contradict each other. This is especially true if the ST contains detailed information on any warnings to the TOE users with regard to the TOE security environment and the security objectives.

777 For guidance on consistency analysis see Annex B.3.

AGD_USR.1.6C

2:AGD_USR.1-6 The evaluator ***shall examine*** the user guidance to determine that it describes all security requirements for the IT environment of the TOE that are relevant to the user.

778 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

779 This work unit relates to IT security requirements only and not to any organisational security policies.

780 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare that with the user guidance to ensure that all security requirements of the ST, that are relevant to the user, are described appropriately in the user guidance.

6.8 Tests activity

781 The purpose of this activity is to determine, by independently testing a subset of the TSF, whether the TOE behaves as specified in the design documentation and in accordance with the TOE security functional requirements specified in the ST.

782 The tests activity at EAL2 contains sub-activities related to the following components:

a) ATE_COV.1;

b) ATE_FUN.1;

c) ATE_IND.2.

6.8.1 Application notes

783 The evaluator analyses the developer's tests to determine the extent to which they are sufficient to demonstrate that security functions perform as specified, and to understand the developer's approach to testing. The evaluator also executes a subset of the developer's tests as documented to gain confidence in the developer's test results. The evaluator will use the results of this analysis as an input to independently testing a subset of the TSF. With respect to this subset, the evaluator's tests take a testing approach that is different from that of the developer's tests, particularly if the developer's tests have shortcomings.

784 Other factors affecting the size and composition of the evaluator's test subset are discussed in the independent testing (ATE_IND.2) sub-activity. One such factor affecting the composition of the subset is *known public domain weaknesses,* information about which the evaluator needs access (e.g. from a scheme).

785 To determine the adequacy of developer's test documentation or to create new tests, the evaluator needs to understand the desired expected behaviour of a security function in the context of the requirements it is to satisfy. The evaluator may choose to focus on one security function of the TSF at a time, examining the ST requirement and the relevant parts of the functional specification and guidance documentation to gain an understanding of the way the TOE is expected to behave.

6.8.2 Evaluation of coverage (ATE_COV.1)

6.8.2.1 Objectives

786 The objective of this sub-activity is to determine whether the developer's test coverage evidence shows correspondence between the tests identified in the test documentation and the functional specification.

6.8.2.2 Application notes

787 The coverage analysis provide by the developer is required to show the correspondence between the test provided as evaluation evidence and the

functional specification. However, the coverage analysis need not demonstrate that all security functions have been tested, or that all external interfaces to the TSF have been tested. Such shortcomings are considered by the evaluator during the independent testing (ATE_IND.2) sub-activity.

6.8.2.3 Input

788 The evaluation evidence for this sub-activity is:

a) the functional specification;

b) the test documentation;

c) the test coverage evidence.

6.8.2.4 Evaluator actions

789 This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_COV.1.1E.

6.8.2.4.1 Action ATE_COV.1.1E

ATE_COV.1.1C

2:ATE_COV.1-1 The evaluator ***shall examine*** the test coverage evidence to determine that the correspondence between the tests identified in the test documentation and the functional specification is accurate.

790 Correspondence may take the form of a table or matrix. The coverage evidence required for this component will reveal the extent of coverage, rather than to show complete coverage. In cases where coverage is shown to be poor the evaluator should increase the level of independent testing to compensate.

791 Figure 6.2 displays a conceptual framework of the correspondence between security functions described in the functional specification and the tests outlined in the test documentation used to test them. Tests may involve one or multiple security functions depending on the test dependencies or the overall goal of the test being performed.

792 The identification of the tests and the security functions presented in the test coverage evidence should be unambiguous, providing a clear correspondence

between the identified tests and the functional specification of the security functions tested.

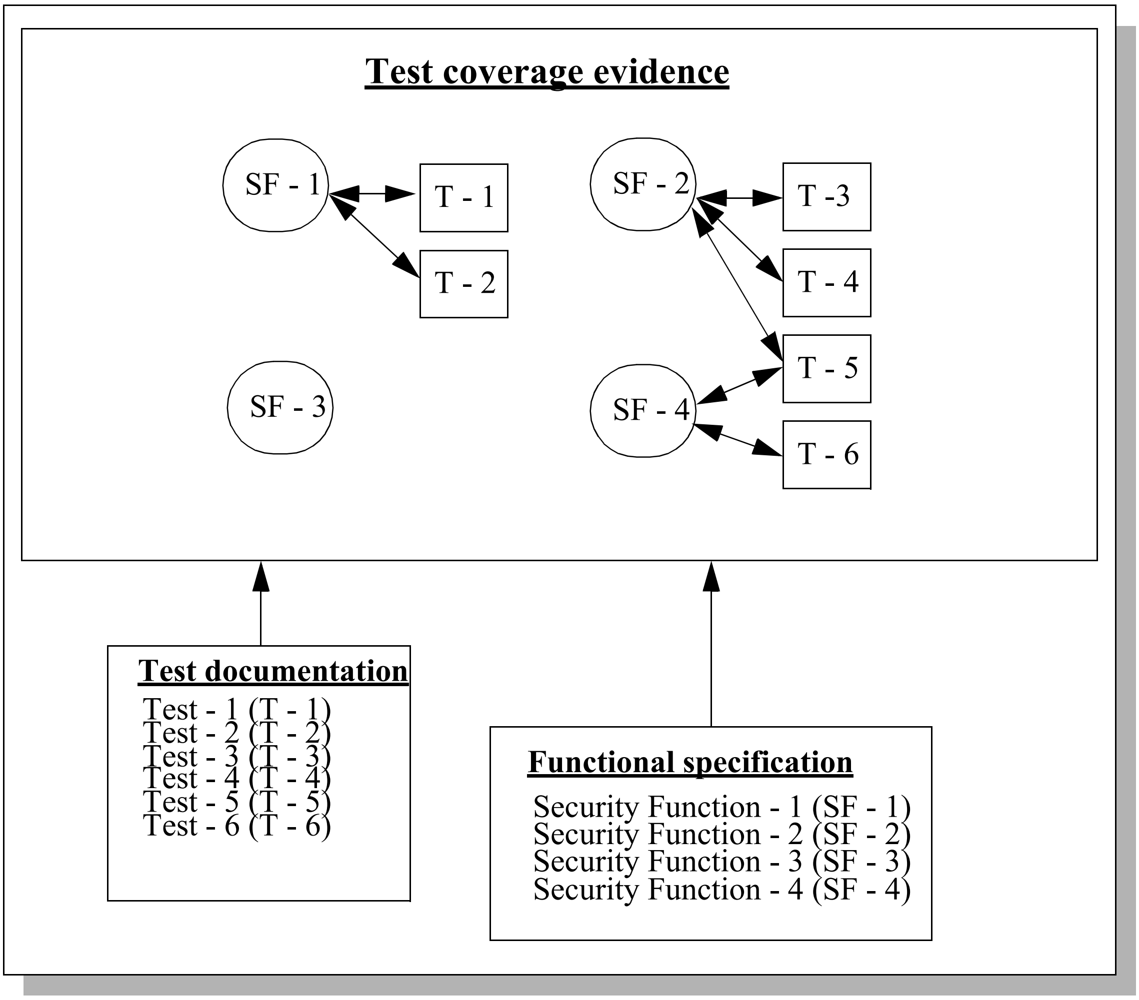

Figure 6.2 A conceptual framework of the test coverage evidence

793 In Figure 6.2 SF-3 does not have tests attributed to it; therefore, coverage with respect to the functional specification is incomplete. Incomplete coverage, however, will not impact the verdict of this sub-activity as the coverage evidence does not have to show complete coverage of the security functions identified in the functional specification.

6.8.3 Evaluation of functional tests (ATE_FUN.1)

6.8.3.1 Objectives

794 The objective of this sub-activity is to determine whether the developer's functional test documentation is sufficient to demonstrate that security functions perform as specified.

6.8.3.2 Application notes

795 The extent to which the test documentation is required to cover the TSF is dependent upon the coverage assurance component.

796 For the developer tests provided, the evaluator determines whether the tests are repeatable, and the extent to which the developer's tests can be used for the evaluator's independent testing effort. Any security function for which the developer's test results indicate that it may not perform as specified should be tested independently by the evaluator to determine whether or not it does.

6.8.3.3 Input

797 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test procedures.

6.8.3.4 Evaluator actions

798 This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_FUN.1.1E.

6.8.3.4.1 Action ATE_FUN.1.1E

ATE_FUN.1.1C

2:ATE_FUN.1-1 The evaluator **shall check** that the test documentation includes test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C

2:ATE_FUN.1-2 The evaluator ***shall check*** that the test plan identifies the security functions to be tested.

799 One method that could be used to identify the security function to be tested is a reference to the appropriate part(s) of the functional specification that specifies the particular security function.

800 The evaluator may wish to employ a sampling strategy when performing this work unit.

801 For guidance on sampling see Annex B.2.

2:ATE_FUN.1-3 The evaluator ***shall examine*** the test plan to determine that it describes the goal of the tests performed.

802 The test plan provides information about how the security functions are tested and the test configuration in which testing occurs.

803 The evaluator may wish to employ a sampling strategy when performing this work unit.

804 For guidance on sampling see Annex B.2.

2:ATE_FUN.1-4 The evaluator ***shall examine*** the test plan to determine that the TOE test configuration is consistent with the configuration identified for evaluation in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

805 ~~The TOE used for testing should have the same unique reference as established by the ACM_CAP.2 sub-activity and the developer supplied test documentation.~~ The TOE referred to in the developer's test plan should have the same unique reference as established by the ACM_CAP.2 sub-activity.

806 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator verifies that there are test configurations identified in the developer test documentation that are consistent with each evaluated configuration described in the ST.

807 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

2:ATE_FUN.1-5 The evaluator ***shall examine*** the test plan to determine that it is consistent with the test procedure descriptions.

808 The evaluator may wish to employ a sampling strategy when performing this work unit.

809 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.3C

2:ATE_FUN.1-6 The evaluator ***shall check*** that the test procedure descriptions identify each security function behaviour to be tested.

810 One method that may be used to identify the security function behaviour to be tested is a reference to the appropriate part(s) of the design specification that specifies the particular behaviour to be tested.

811 The evaluator may wish to employ a sampling strategy when performing this work unit.

812 For guidance on sampling see Annex B.2.

2:ATE_FUN.1-7 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to establish reproducible initial test conditions including ordering dependencies if any.

813 Some steps may have to be performed to establish initial conditions. For example, user accounts need to be added before they can be deleted. An example of ordering dependencies on the results of other tests is the need to test the audit function before relying on it to produce audit records for another security mechanism such as access control. Another example of an ordering dependency would be where one test case generates a file of data to be used as input for another test case.

814 The evaluator may wish to employ a sampling strategy when performing this work unit.

815 For guidance on sampling see Annex B.2.

2:ATE_FUN.1-8 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to have a reproducible means to stimulate the security functions and to observe their behaviour.

816 Stimulus is usually provided to a security function externally through the TSFI. Once an input (stimulus) is provided to the TSFI, the behaviour of the security function can then be observed at the TSFI. Reproducibility is not assured unless the test procedures contain enough detail to unambiguously describe the stimulus and the behaviour expected as a result of this stimulus.

817 The evaluator may wish to employ a sampling strategy when performing this work unit.

818 For guidance on sampling see Annex B.2.

2:ATE_FUN.1-9 The evaluator ***shall examine*** the test procedure descriptions to determine that they are consistent with the test procedures.

819 If the test procedure descriptions are the test procedures, then this work unit is not applicable and is therefore considered to be satisfied.

820 The evaluator may wish to employ a sampling strategy when performing this work unit.

821 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.4C

2:ATE_FUN.1-10 The evaluator ***shall examine*** the test documentation to determine that sufficient expected tests results are included.

822 The expected test results are needed to determine whether or not a test has been successfully performed. Expected test results are sufficient if they are unambiguous and consistent with expected behaviour given the testing approach.

823 The evaluator may wish to employ a sampling strategy when performing this work unit.

824 For guidance on sampling see Annex B.2.

ATE_FUN.1.5C

2:ATE_FUN.1-11 The evaluator ***shall check*** that the expected test results in the test documentation are consistent with the actual test results provided.

825 A comparison of the actual and expected test results provided by the developer will reveal any inconsistencies between the results.

826 It may be that a direct comparison of actual results cannot be made until some data reduction or synthesis has been first performed. In such cases, the developer's test documentation should describe the process to reduce or synthesize the actual data.

827 For example, the developer may need to test the contents of a message buffer after a network connection has occurred to determine the contents of the buffer. The message buffer will contain a binary number. This binary number would have to be converted to another form of data representation in order to make the test more meaningful. The conversion of this binary representation of data into a higher-level representation will have to be described by the developer in enough detail to allow an evaluator to perform the conversion process (i.e. synchronous or asynchronous transmission, number of stop bits, parity, etc.).

828 It should be noted that the description of the process used to reduce or synthesize the actual data is used by the evaluator not to actually perform the necessary modification but to assess whether this process is correct. It is up to the developer to transform the expected test results into a format that allows an easy comparison with the actual test results.

829 The evaluator may wish to employ a sampling strategy when performing this work unit.

830 For guidance on sampling see Annex B.2.

831 If the expected and actual test results for any test are not the same, then a demonstration of the correct operation of a security function has not been achieved. Such an occurrence will influence the evaluator's independent testing effort to include testing the implicated security function. The evaluator should also consider increasing the sample of evidence upon which this work unit is performed.

2:ATE_FUN.1-12 The evaluator ***shall report*** the developer testing effort, outlining the testing approach, configuration, depth and results.

832 The developer testing information recorded in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing of the TOE by the developer. The intent of providing this information is to give a meaningful overview of the developer testing effort. It is not intended that the information regarding developer testing in the ETR be an exact reproduction of specific test steps or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the developer's testing approach, amount of testing performed, TOE test configurations, and the overall results of the developer testing.

833 Information that would typically be found in the ETR section regarding the developer testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested;

b) testing approach. An account of the overall developer testing strategy employed;

c) amount of developer testing performed. A description on the extent of coverage and depth of developer testing;

d) testing results. A description of the overall developer testing results.

834 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the developer testing effort.

6.8.4 Evaluation of independent testing (ATE_IND.2)

6.8.4.1 Objectives

835 The purpose of this activity is to determine, by independently testing a subset of the TSF, whether the TOE behaves as specified, and to gain confidence in the developer's test results by performing a sample of the developer's tests.

6.8.4.2 Input

836 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures;

f) the test documentation;

g) the test coverage analysis;

h) the TOE suitable for testing.

6.8.4.3 Evaluator actions

837 This sub-activity comprises three CC Part 3 evaluator action elements:

a) ATE_IND.2.1E;

b) ATE_IND.2.2E;

c) ATE_IND.2.3E.

6.8.4.3.1 Action ATE_IND.2.1E

ATE_IND.2.1C

2:ATE_IND.2-1 The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

838 The TOE used for evaluator testing should have the same unique reference as established by the ACM_CAP.2 sub-activity ~~and the developer supplied test documentation~~.

839 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator's TOE ~~verifies that there are~~ test configurations should be consistent with each evaluated configuration described in the ST.

840 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

841 If any test resources are used (e.g. meters, analysers) it will be the evaluator's responsibility to ensure that these resources are calibrated correctly.

2:ATE_IND.2-2 The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

842 It is possible for the evaluator to determine the state of the TOE in a number of ways. For example, previous successful completion of the ADO_IGS.1 sub-activity will satisfy this work unit if the evaluator still has confidence that the TOE being used for testing was installed properly and is in a known state. If this is not the case, then the evaluator should follow the developer's procedures to install, generate and start up the TOE, using the supplied guidance only.

843 If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed could satisfy work unit 2:ADO_IGS.1-2.

ATE_IND.2.2C

2:ATE_IND.2-3 The evaluator ***shall examine*** the set of resources provided by the developer to determine that they are equivalent to the set of resources used by the developer to functionally test the TSF.

844 The resource set may include laboratory access and special test equipment, among others. Resources that are not identical to those used by the developer need to be equivalent in terms of any impact they may have on test results.

6.8.4.3.2 Action ATE_IND.2.2E

2:ATE_IND.2-4 The evaluator ***shall devise*** a test subset.

845 The evaluator selects a test subset and testing strategy that is appropriate for the TOE. One extreme testing strategy would be to have the test subset contain as many security functions as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few security functions based on their perceived relevance and rigorously test these functions.

846 Typically the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the security functional requirements identified in the ST using at least one test, but testing need not demonstrate exhaustive specification testing.

847 The evaluator, when selecting the subset of the TSF to be tested, should consider the following factors:

a) The developer test evidence. The developer test evidence consists of: the test coverage analysis, and the test documentation. The developer test evidence will provide insight as to how the security functions have been exercised by the developer during testing. The evaluator applies this information when developing new tests to independently test the TOE. Specifically the evaluator should consider:

   1) augmentation of developer testing for specific security function(s). The evaluator may wish to perform more of the same type of tests by varying parameters to more rigorously test the security function.

   2) supplementation of developer testing strategy for specific security function(s). The evaluator may wish to vary the testing approach of a specific security function by testing it using another test strategy.

b) The number of security functions from which to draw upon for the test subset. Where the TOE includes only a small number of security functions, it may be practical to rigourously test all of the security functions. For TOEs with a large number of security functions this will not be cost-effective, and sampling is required.

c) Maintaining a balance of evaluation activities. The evaluator effort expended on the test activity should be commensurate with that expended on any other evaluation activity. Given that the requirements in ATE_COV.1 allow for significant variation in the level of test coverage provided by the developer, the level of coverage provided will be a significant factor in determining the appropriate effort expended by the evaluator.

848 The evaluator selects the security functions to compose the subset. This selection will depend on a number of factors, and consideration of these factors may also influence the choice of test subset size:

a) Rigour of developer testing of the security functions. Some security functions identified in the functional specification may have had little or no developer test evidence attributed to them. Those security functions that the evaluator determines require additional testing should be included in the test subset.

b) Developer test results. If the results of developer tests cause the evaluator to doubt that a security function, or aspect thereof, operates as specified, then the evaluator should include such security functions in the test subset.

c) Known public domain weaknesses commonly associated with the type of TOE (e.g. operating system, firewall). Know public domain weaknesses associated with the type of TOE will influence the selection process of the test subset. The evaluator should include those security functions that address known public domain weaknesses for that type of TOE in the subset (know public domain weaknesses in this context does not refer to vulnerabilities as such but to inadequacies or problem areas that have been experienced with this particular type of TOE). If no such weaknesses are known, then a more general approach of selecting a broad range of security functions may be more appropriate.

d) Significance of security functions. Those security functions more significant than others in terms of the security objectives for the TOE should be included in the test subset.

e) SOF claims made in the ST. All security functions for which a specific SOF claim has been made should be included in the test subset.

f) Complexity of the security function. Complex security functions may require complex tests that impose onerous requirements on the developer or evaluator, which will not be conducive to cost-effective evaluations. Conversely, complex security functions are a likely area to find errors and are good candidates for the subset. The evaluator will need to strike a balance between these considerations.

g) Implicit testing. Testing some security functions may often implicitly test other security functions, and their inclusion in the subset may maximize the number of security functions tested (albeit implicitly). Certain interfaces will typically be used to provide a variety of security functionality, and will tend to be the target of an effective testing approach.

h) Types of interfaces to the TOE (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

i) Functions that are innovative or unusual. Where the TOE contains innovative or unusual security functions, which may feature strongly in marketing literature, these should be strong candidates for testing.

849 This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

850 For guidance on sampling see Annex B.2.

2:ATE_IND.2-5 The evaluator ***shall produce*** test documentation for the test subset that is sufficiently detailed to enable the tests to be reproducible.

851 With an understanding of the expected behaviour of a security function, from the ST and the functional specification, the evaluator has to determine the most feasible way to test the function. Specifically the evaluator considers:

a) the approach that will be used, for instance, whether the security function will be tested at an external interface, at an internal interface using a test harness, or will an alternate test approach be employed (e.g. in exceptional circumstances, a code inspection);

b) the security function interface(s) that will be used to stimulate the security function and observe responses;

c) the initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

d) special test equipment that will be required to either stimulate a security function (e.g. packet generators) or make observations of a security function (e.g. network analysers).

852 The evaluator may find it practical to test each security function using a series of test cases, where each test case will test a very specific aspect of expected behaviour.

853 The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant design specification, and to the ST, if necessary.

2:ATE_IND.2-6 The evaluator ***shall conduct*** testing.

854 The evaluator uses the test documentation developed as a basis for executing tests on the TOE. The test documentation is used as a basis for testing but this does not preclude the evaluator from performing additional ad hoc tests. The evaluator may devise new tests based on behaviour of the TOE discovered during testing. These new tests are recorded in the test documentation.

2:ATE_IND.2-7 The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the security function behaviour to be tested;

b) instructions to connect and setup all required test equipment as required to conduct the test;

c) instructions to establish all prerequisite test conditions;

d) instructions to stimulate the security function;

e) instructions for observing the behaviour of the security function;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE;

h) actual test results.

855 The level of detail should be such that another evaluator could repeat the tests and obtain an equivalent result. While some specific details of the test results may be different (e.g. time and date fields in an audit record) the overall result should be identical.

856 There may be instances when it is unnecessary to provide all the information presented in this work unit (e.g. the actual test results of a test may not require any analysis before a comparison between the expected results can be made). The determination to omit this information is left to the evaluator, as is the justification.

2:ATE_IND.2-8 The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

857 Any differences in the actual and expected test results may indicate that the TOE does not perform as specified or that the evaluator test documentation may be incorrect. Unexpected actual results may require corrective maintenance to the TOE or test documentation and perhaps require re-running of impacted tests and modifying the test sample size and composition. This determination is left to the evaluator, as is its justification.

6.8.4.3.3 Action ATE_IND.2.3E

2:ATE_IND.2-9 The evaluator ***shall conduct*** testing using a sample of tests found in the developer test plan and procedures.

858 The overall aim of this work unit is to perform a sufficient number of the developer tests to confirm the validity of the developer's test results. The evaluator has to decide on the size of the sample, and the developer tests that will compose the sample.

859 Taking into consideration the overall evaluator effort for the entire tests activity, normally 20% of the developer's tests should be performed although this may vary according to the nature of the TOE, and the test evidence supplied.

860 All the developer tests can be traced back to specific security function(s). Therefore, the factors to consider in the selection of the tests to compose the sample are similar to those listed for subset selection in work-unit ATE_IND.2-4. Additionally, the evaluator may wish to employ a random sampling method to select developer tests to include in the sample.

861 For guidance on sampling see Annex B.2.

2:ATE_IND.2-10 The evaluator ***shall check*** that all the actual test results are consistent with the expected test results.

862 Inconsistencies between the developer's expected test results and actual test results will compel the evaluator to resolve the discrepancies. Inconsistencies encountered by the evaluator could be resolved by a valid explanation and resolution of the inconsistencies by the developer.

863 If a satisfactory explanation or resolution can not be reached, the evaluator's confidence in the developer's test results may be lessened and it may even be necessary for the evaluator to increase the sample size, to regain confidence in the developer testing. If the increase in sample size does not satisfy the evaluator's concerns, it may be necessary to repeat the entire set of developer's tests. Ultimately, to the extent that the TSF subset identified in work unit ATE_IND.2-4 is adequately tested, deficiencies with the developer's tests need to result in either corrective action to the developer's tests or in the production of new tests by the evaluator.

2:ATE_IND.2-11 The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

864 The evaluator testing information reported in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing activity during the evaluation. The intent of providing this information is to give a meaningful overview of the testing effort. It is not intended that the information regarding testing in the ETR be an exact reproduction of specific test instructions or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the testing approach chosen, amount of evaluator testing performed, amount of developer tests performed, TOE test configurations, and the overall results of the testing activity.

865 Information that would typically be found in the ETR section regarding the evaluator testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested.

b) subset size chosen. The amount of security functions that were tested during the evaluation and a justification for the size.

c) selection criteria for the security functions that compose the subset. Brief statements about the factors considered when selecting security functions for inclusion in the subset.

d) security functions tested. A brief listing of the security functions that merited inclusion in the subset.

e) developer tests performed. The amount of developer tests performed and a brief description of the criteria used to select the tests.

f) verdict for the activity. The overall judgement on the results of testing during the evaluation.

866 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the testing the evaluator performed during the evaluation.

6.9 Vulnerability assessment activity

867 The purpose of the vulnerability assessment activity is to determine the exploitability of flaws or weaknesses in the TOE in the intended environment. This determination is based upon analysis performed by the developer, and is supported by evaluator penetration testing.

868 The vulnerability assessment activity at EAL2 contains sub-activities related to the following components:

a) AVA_SOF.1;

b) AVA_VLA.1.

6.9.1 Evaluation of strength of TOE security functions (AVA_SOF.1)

6.9.1.1 Objectives

869 The objectives of this sub-activity are to determine whether SOF claims are made in the ST for all probabilistic or permutational mechanisms and whether the developer's SOF claims made in the ST are supported by an analysis that is correct.

6.9.1.2 Application notes

870 SOF analysis is performed on mechanisms that are probabilistic or permutational in nature, such as password mechanisms or biometrics. Although cryptographic mechanisms are also probabilistic in nature and are often described in terms of *strength*, AVA_SOF.1 is not applicable to cryptographic mechanisms. For such mechanisms, the evaluator should seek scheme guidance.

871 Although SOF analysis is performed on the basis of individual mechanisms, the overall determination of SOF is based on functions. Where more than one probabilistic or permutational mechanism is employed to provide a security function, each distinct mechanism must be analysed. The manner in which these mechanisms combine to provide a security function will determine the overall SOF level for that function. The evaluator needs design information to understand how the mechanisms work together to provide a function, and a minimum level for such information is given by the dependency on ADV_HLD.1. The actual design information available to the evaluator is determined by the EAL, and the available information should be used to support the evaluator's analysis when required.

872 For a discussion on SOF in relation to multiple TOE domains see Section 4.4.6.

6.9.1.3 Input

873 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the strength of TOE security functions analysis.

6.9.1.4 Evaluator actions

874 This sub-activity comprises two CC Part 3 evaluator action elements:

a) AVA_SOF.1.1E;

b) AVA_SOF.1.2E.

6.9.1.4.1 Action AVA_SOF.1.1E

AVA_SOF.1.1C

2:AVA_SOF.1-1 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a SOF rating.

875 If SOF claims are expressed solely as SOF metrics, then this work unit is not applicable and is therefore considered to be satisfied.

876 A SOF rating is expressed as one of SOF-basic, SOF-medium or SOF-high, which are defined in terms of attack potential - refer to the CC Part 1 Glossary. A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational security mechanisms. However, individual mechanisms may have a SOF claim expressed as a rating that exceeds the overall SOF requirement.

877 Guidance on determining the attack potential necessary to effect an attack and, hence, to determine SOF as a rating is in Annex B.8.

878 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.2C

2:AVA_SOF.1-2 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a metric.

879 If SOF claims are expressed solely as SOF ratings, then this work unit is not applicable and is therefore considered to be satisfied.

880 A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational mechanisms. However, individual mechanisms may have a SOF claim expressed as a metric that meets or exceeds the overall SOF requirement.

881 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.1C and AVA_SOF.1.2C

2:AVA_SOF.1-3 The evaluator ***shall examine*** the SOF analysis to determine that any assertions or assumptions supporting the analysis are valid.

882 For example, it may be a flawed assumption that a particular implementation of a pseudo-random number generator will possess the required entropy necessary to seed the security mechanism to which the SOF analysis is relevant.

883 Assumptions supporting the SOF analysis should reflect the *worst case*, unless *worst case* is invalidated by the ST. Where a number of different possible scenarios exist, and these are dependent on the behaviour of the human user or attacker, the case that represents the lowest strength should be assumed unless, as previously stated, this case is invalid.

884 For example, a strength claim based upon a maximum theoretical password space (i.e. all printable ASCII characters) would not be *worst case* because it is human behaviour to use natural language passwords, effectively reducing the password space and associated strength. However, such an assumption could be appropriate if the TOE used IT measures, identified in the ST, such as password filters to minimise the use of natural language passwords.

2:AVA_SOF.1-4 The evaluator ***shall examine*** the SOF analysis to determine that any algorithms, principles, properties and calculations supporting the analysis are correct.

885 The nature of this work unit is highly dependent upon the type of mechanism being considered. Annex B.8 provides an example SOF analysis for an identification and authentication function that is implemented using a password mechanism; the analysis considers the maximum password space to ultimately arrive at a SOF rating. For biometrics, the analysis should consider resolution and other factors impacting the mechanism's susceptibility to spoofing.

886 SOF expressed as a rating is based on the minimum attack potential required to defeat the security mechanism. The SOF ratings are defined in terms of attack potential in CC Part 1 Glossary.

887 For guidance on attack potential see Annex B.8.

2:AVA_SOF.1-5 The evaluator ***shall examine*** the SOF analysis to determine that each SOF claim is met or exceeded.

888 For guidance on the rating of SOF claims see Annex B.8.

2:AVA_SOF.1-6 The evaluator ***shall examine*** the SOF analysis to determine that all functions with a SOF claim meet the minimum strength level defined in the ST.

6.9.1.4.2 Action AVA_SOF.1.2E

2:AVA_SOF.1-7 The evaluator ***shall examine*** the functional specification, the high-level design, the user guidance and the administrator guidance to determine that all probabilistic or permutational mechanisms have a SOF claim.

889 The identification by the developer of security functions that are realised by probabilistic or permutational mechanisms is verified during the ST evaluation. However, because the TOE summary specification may have been the only evidence available upon which to perform that activity, the identification of such mechanisms may be incomplete. Additional evaluation evidence required as input to this sub-activity may identify additional probabilistic or permutational mechanisms not already identified in the ST. If so, the ST will have to be updated appropriately to reflect the additional SOF claims and the developer will need to provide additional analysis that justifies the claims as input to evaluator action AVA_SOF.1.1E.

2:AVA_SOF.1-8 The evaluator ***shall examine*** the SOF claims to determine that they are correct.

890 Where the SOF analysis includes assertions or assumptions (e.g. about how many authentication attempts are possible per minute), the evaluator should independently confirm that these are correct. This may be achieved through testing or through independent analysis.

6.9.2 Evaluation of vulnerability analysis (AVA_VLA.1)

6.9.2.1 Objectives

891 The objective of this sub-activity is to determine whether the TOE, in its intended environment, has exploitable obvious vulnerabilities.

6.9.2.2 Application notes

892 The use of the term *guidance* in this sub-activity refers to the user guidance, the administrator guidance, and the secure installation, generation, and start-up procedures.

893 The consideration of exploitable vulnerabilities will be determined by the security objectives and functional requirements in the ST. For example, if measures to prevent bypass of the security functions are not required in the ST (FPT_PHP, FPT_RVM and FPT_SEP are absent) then vulnerabilities based on bypass should not be considered.

894 Vulnerabilities may be in the public domain, or not, and may require skill to exploit, or not. These two aspects are related, but are distinct. It should not be assumed that, simply because a vulnerability is in the public domain, it can be easily exploited.

895 The following terms are used in the guidance with specific meaning:

a) Vulnerability - a weakness in the TOE that can be used to violate a security policy in some environment;

b) Vulnerability analysis - A systematic search for vulnerabilities in the TOE, and an assessment of those found to determine their relevance for the intended environment for the TOE;

c) Obvious vulnerability - a vulnerability that is open to exploitation that requires a minimum of understanding of the TOE, technical sophistication and resources;

d) Potential vulnerability - A vulnerability the existence of which is suspected (by virtue of a postulated attack path), but not confirmed, in the TOE;

e) Exploitable vulnerability - A vulnerability that can be exploited in the intended environment for the TOE;

f) Non-exploitable vulnerability - A vulnerability that cannot be exploited in the intended environment for the TOE;

g) Residual vulnerability - A non-exploitable vulnerability that could be exploited by an attacker with greater attack potential than is anticipated in the intended environment for the TOE;

h) Penetration testing - Testing carried out to determine the exploitability of identified TOE potential vulnerabilities in the intended environment for the TOE.

6.9.2.3 Input

896 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures;

g) the vulnerability analysis;

h) the strength of function claims analysis;

i) the TOE suitable for testing.

897 Other input for this sub-activity is:

a) current information regarding obvious vulnerabilities (e.g. from an overseer).

6.9.2.4 Evaluator actions

898 This sub-activity comprises two CC Part 3 evaluator action elements:

a) AVA_VLA.1.1E;

b) AVA_VLA.1.2E.

6.9.2.4.1 Action AVA_VLA.1.1E

Interp Note : The following citation is changed as a result of Interpretation 051.

AVA_VLA.1.1C, AVA_VLA.1.2C and AVA_VLA.1.3C

2:AVA_VLA.1-1 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that the search for obvious vulnerabilities has considered all relevant information.

899 The developer's vulnerability analysis should cover the developer's search for obvious vulnerabilities in at least all evaluation deliverables and public domain

information sources. The evaluator should use the evaluation deliverables, not to perform an independent vulnerability analysis (not required at AVA_VLA.1), but as a basis for assessing the developer's search for obvious vulnerabilities.

Interp Note : The following paragraph is added as a result of Interpretation 031.

Information in the public domain is highly dynamic. Therefore, it is possible that new vulnerabilities are reported in the public domain between the time the developer performs the vulnerability analysis and the time that the evaluation is completed. The point at which monitoring of the public domain information ceases is an evaluation authority issue; therefore guidance and agreement should be sought from the evaluation authority.

2:AVA_VLA.1-2 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that each obvious vulnerability is described and that a rationale is given for why it is not exploitable in the intended environment for the TOE.

900 The developer is expected to search for obvious vulnerabilities, based on knowledge of the TOE, and of public domain information sources. Given the requirement to identify only obvious vulnerabilities, a detailed analysis is not expected. The developer filters this information, based on the above definition, and shows that obvious vulnerabilities are not exploitable in the intended environment.

901 The evaluator needs to be concerned with three aspects of the developer's analysis:

a) whether the developer's analysis has considered all evaluation deliverables;

b) whether appropriate measures are in place to prevent the exploitation of obvious vulnerabilities in the intended environment;

c) whether some obvious vulnerabilities remain unidentified.

902 The evaluator should not be concerned over whether identified vulnerabilities are obvious or not, unless this is used by the developer as a basis for determining non-exploitability. In such a case the evaluator validates the assertion by determining resistance to an attacker with low attack potential for the identified vulnerability.

903 The concept of *obvious vulnerabilities* is not related to that of *attack potential*. The latter is determined by the evaluator during independent vulnerability analysis. Since this activity is not performed for AVA_VLA.1, there is normally no searching and filtering by the evaluator on the basis of attack potential. However, the evaluator may still discover potential vulnerabilities during the evaluation, and the determination of how these should be addressed will be made by reference to the definition of obvious vulnerabilities and the concept of low attack potential.

904 The determination as to whether some obvious vulnerabilities remain unidentified is limited to assessment of the validity of the developer's analysis, a comparison with available public domain vulnerability information, and a comparison with any further vulnerabilities identified by the evaluator during the course of other evaluation activities.

905 A vulnerability is termed non-exploitable if one or more of the following conditions exist:

a) security functions or measures in the (IT or non-IT) environment prevent exploitation of the vulnerability in the intended environment. For instance, restricting physical access to the TOE to authorised users only may effectively render a TOE's vulnerability to tampering unexploitable;

b) the vulnerability is exploitable but only by attackers possessing moderate or high attack potential. For instance, a vulnerability of a distributed TOE to session hijack attacks requires an attack potential beyond that required to exploit an obvious vulnerability. However, such vulnerabilities are reported in the ETR as residual vulnerabilities.

c) either the threat is not claimed to be countered or the violable organisational security policy is not claimed to be achieved by the ST. For instance, a firewall whose ST makes no availability policy claim and is vulnerable to TCP SYN attacks (an attack on a common Internet protocol that renders hosts incapable of servicing connection requests) should not fail this evaluator action on the basis of this vulnerability alone.

906 For guidance on determining attack potential necessary to exploit a vulnerability see Annex B.8.

2:AVA_VLA.1-3 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that it is consistent with the ST and the guidance.

907 The developer's vulnerability analysis may address a vulnerability by suggesting specific configurations or settings for TOE functions. If such operating constraints are deemed to be effective and consistent with the ST, then all such configurations/ settings should be adequately described in the guidance so that they may be employed by the consumer.

6.9.2.4.2 Action AVA_VLA.1.2E

2:AVA_VLA.1-4 The evaluator ***shall devise*** penetration tests, building on the developer vulnerability analysis.

908 The evaluator prepares for penetration testing:

a) as necessary to attempt to disprove the developer's analysis in cases where the developer's rationale for why a vulnerability is unexploitable is suspect in the opinion of the evaluator;

b) as necessary to determine the susceptibility of the TOE, in its intended environment, to an obvious vulnerability not considered by the developer. The evaluator should have access to current information (e.g. from the overseer) regarding obvious public domain vulnerabilities that may not have been considered by the developer, and may also have identified potential vulnerabilities as a result of performing other evaluation activities.

909 The evaluator is not expected to test for vulnerabilities (including those in the public domain) beyond those which are obvious. In some cases, however, it will be necessary to carry out a test before the exploitability can be determined. Where, as a result of evaluation expertise, the evaluator discovers a vulnerability that is beyond obvious, this is reported in the ETR as a residual vulnerability.

910 With an understanding of the suspected obvious vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically the evaluator considers:

a) the security function interfaces that will be used to stimulate the TSF and observe responses;

b) initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

c) special test equipment that will be required to either stimulate a security function or make observations of a security function (although it is unlikely that specialist equipment would be required to exploit an obvious vulnerability).

911 The evaluator will probably find it practical to carry out penetration testing using a series of test cases, where each test case will test for a specific obvious vulnerability.

2:AVA_VLA.1-5 The evaluator ***shall produce*** penetration test documentation for the tests that build upon the developer vulnerability analysis, in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the obvious vulnerability the TOE is being tested for;

b) instructions to connect and setup all required test equipment as required to conduct the penetration test;

c) instructions to establish all penetration test prerequisite initial conditions;

d) instructions to stimulate the TSF;

e) instructions for observing the behaviour of the TSF;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE.

912 The intent of specifying this level of detail in the test documentation is to allow another evaluator to repeat the tests and obtain an equivalent result.

2:AVA_VLA.1-6 The evaluator ***shall conduct*** penetration testing, building on the developer vulnerability analysis.

913 The evaluator uses the penetration test documentation resulting from work unit 2:AVA_VLA.1-4 as a basis for executing penetration tests on the TOE, but this does not preclude the evaluator from performing additional ad hoc penetration tests. If required, the evaluator may devise ad hoc tests as a result of information learned during penetration testing that, if performed by the evaluator, are to be recorded in the penetration test documentation. Such tests may be required to follow up unexpected results or observations, or to investigate potential vulnerabilities suggested to the evaluator during the pre-planned testing.

2:AVA_VLA.1-7 The evaluator ***shall record*** the actual results of the penetration tests.

914 While some specific details of the actual test results may be different from those expected (e.g. time and date fields in an audit record) the overall result should be identical. Any differences should be justified.

2:AVA_VLA.1-8 The evaluator ***shall examine*** the results of all penetration testing and the conclusions of all vulnerability analysis to determine that the TOE, in its intended environment, has no exploitable obvious vulnerabilities.

915 If the results reveal that the TOE has obvious vulnerabilities, exploitable in its intended environment, then this results in a failed verdict for the evaluator action.

2:AVA_VLA.1-9 The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

916 The penetration testing information reported in the ETR allows the evaluator to convey the overall penetration testing approach and effort expended on this sub-activity. The intent of providing this information is to give a meaningful overview of the evaluator's penetration testing effort. It is not intended that the information regarding penetration testing in the ETR be an exact reproduction of specific test steps or results of individual penetration tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the penetration testing approach chosen, amount of penetration testing performed, TOE test configurations, and the overall results of the penetration testing activity.

917 Information that would typically be found in the ETR section regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) security functions penetration tested. A brief listing of the security functions that were the focus of the penetration testing;

c) verdict for the sub-activity. The overall judgement on the results of penetration testing.

918 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the penetration testing the evaluator performed during the evaluation.

2:AVA_VLA.1-10 The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. CEM activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the implicated security function(s), objective(s) not met, organisational security policy(ies) contravened and threat(s) realised;

c) a description;

d) whether it is exploitable in its intended environment or not (i.e. exploitable or residual);

e) identification of evaluation party (e.g. developer, evaluator) who identified it.

Chapter 7

EAL3 evaluation

7.1 Introduction

919 EAL3 provides a moderate level of assurance. The security functions are analysed using a functional specification, guidance documentation, and the high-level design of the TOE to understand the security behaviour. The analysis is supported by independent testing of a subset of the TOE security functions, evidence of developer testing based on the functional specification and the high level design, selective confirmation of the developer test results, analysis of strengths of the functions, and evidence of a developer search for obvious vulnerabilities. Further assurance is gained through the use of development environment controls, TOE configuration management, and evidence of secure delivery procedures.

7.2 Objectives

920 The objective of this chapter is to define the minimal evaluation effort for achieving an EAL3 evaluation and to provide guidance on ways and means of accomplishing the evaluation.

7.3 EAL3 evaluation relationships

921 An EAL3 evaluation covers the following:

a) evaluation input task (Chapter 2);

b) EAL3 evaluation activities comprising the following:

    1) evaluation of the ST (Chapter 4);

    2) evaluation of the configuration management (Section 7.4);

    3) evaluation of the delivery and operation documents (Section 7.5);

    4) evaluation of the development documents (Section 7.6);

    5) evaluation of the guidance documents (Section 7.7);

    6) evaluation of the life cycle support (Section 7.8);

    7) evaluation of the tests (Section 7.9);

    8) testing (Section 7.9);

9) evaluation of the vulnerability assessment (Section 7.10);

c) evaluation output task (Chapter 2).

922 The evaluation activities are derived from the EAL3 assurance requirements contained in the CC Part 3.

923 The ST evaluation is started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities.

924 The sub-activities comprising an EAL3 evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator.

925 For guidance on dependencies see Annex B.4.

7.4 Configuration management activity

926 The purpose of the configuration management activity is to assist the consumer in identifying the evaluated TOE, to ensure that configuration items are uniquely identified, and to ensure the adequacy of the procedures that are used by the developer to control and track changes that are made to the TOE. This includes details on what changes are tracked, and how potential changes are incorporated.

927 The configuration management activity at EAL3 contains sub-activities related to the following components:

a) ACM_CAP.3;

b) ACM_SCP.1.

7.4.1 Evaluation of CM capabilities (ACM_CAP.3)

7.4.1.1 Objectives

928 The objectives of this sub-activity are to determine whether the developer has clearly identified the TOE and its associated configuration items, and whether the ability to modify these items is properly controlled.

7.4.1.2 Input

929 The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

7.4.1.3 Evaluator actions

930 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_CAP.3.1E;

7.4.1.3.1 Action ACM_CAP.3.1E

ACM_CAP.3.1C

3:ACM_CAP.3-1 The evaluator ***shall check*** that the version of the TOE provided for evaluation is uniquely referenced.

931 The evaluator should use the developer's CM system to validate the uniqueness of the reference by checking the configuration list to ensure that the configuration items are uniquely identified. Evidence that the version provided for evaluation is uniquely referenced may be incomplete if only one version is examined during the

evaluation, and the evaluator should look for a referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates). However, the absence of any reference will normally lead to a fail verdict against this requirement unless the evaluator is confident that the TOE can be uniquely identified.

932 The evaluator should seek to examine more than one version of the TOE (e.g. during rework following discovery of a vulnerability), to check that the two versions are referenced differently.

ACM_CAP.3.2C

3:ACM_CAP.3-2 The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

933 The evaluator should ensure that the TOE contains a unique reference such that it is possible to distinguish different versions of the TOE. This could be achieved through labelled packaging or media, or by a label displayed by the operational TOE. This is to ensure that it would be possible for consumers to identify the TOE (e.g. at the point of purchase or use).

934 The TOE may provide a method by which it can be easily identified. For example, a software TOE may display its name and version number during the start up routine, or in response to a command line entry. A hardware or firmware TOE may be identified by a part number physically stamped on the TOE.

3:ACM_CAP.3-3 The evaluator ***shall check*** that the TOE references used are consistent.

935 If the TOE is labelled more than once then the labels have to be consistent. For example, it should be possible to relate any labelled guidance documentation supplied as part of the TOE to the evaluated operational TOE. This ensures that consumers can be confident that they have purchased the evaluated version of the TOE, that they have installed this version, and that they have the correct version of the guidance to operate the TOE in accordance with its ST. The evaluator can use the configuration list that is part of the provided CM documentation to verify the consistent use of identifiers.

936 The evaluator also verifies that the TOE reference is consistent with the ST.

937 For guidance on consistency analysis see Annex B.3.

ACM_CAP.3.3C

3:ACM_CAP.3-4 The evaluator ***shall check*** that the CM documentation provided includes a configuration list.

938 A configuration list identifies the items being maintained under configuration control.

Interp Note : The following element is added as a result of Interpretation 003.

ACM_CAP.3.<new C&P element resulting from Interp>C

3:ACM_CAP.2-new The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.

The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.

3:ACM_CAP.3-5 The evaluator ***shall check*** that the CM documentation provided includes a CM plan.

ACM_CAP.3.4C

3:ACM_CAP.3-6 The evaluator ***shall examine*** the configuration list to determine that it identifies the configuration items that comprise the TOE.

939 The minimum scope of configuration items to be covered in the configuration list is given by ACM_SCP.

ACM_CAP.3.5C

3:ACM_CAP.3-7 The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

ACM_CAP.3.6C

Interp Note : The following element is replaced as a result of Interpretation 003.

3:ACM_CAP.3-8 ~~The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.~~ The evaluator shall examine the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

940 ~~The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.~~Assurance that the CM system uniquely identifies all configuration items is gained by examining the identifiers for the configuration items. For both configuration items that comprise the TOE, and drafts of configuration items that are submitted by the developer as evaluation evidence, the evaluator confirms that each configuration item possesses a unique identifier in a manner consistent with the unique identification method that is described in the CM documentation.

ACM_CAP.3.7C

3:ACM_CAP.3-9 The evaluator ***shall examine*** the CM plan to determine that it describes how the CM system is used to maintain the integrity of the TOE configuration items.

941 The descriptions contained in a CM plan may include:

a) all activities performed in the TOE development environment that are subject to configuration management procedures (e.g. creation, modification or deletion of a configuration item);

b) the roles and responsibilities of individuals required to perform operations on individual configuration items (different roles may be identified for different types of configuration item (e.g. design documentation or source code));

c) the procedures that are used to ensure that only authorised individuals can make changes to configuration items;

d) the procedures that are used to ensure that concurrency problems do not occur as a result of simultaneous changes to configuration items;

e) the evidence that is generated as a result of application of the procedures. For example, for a change to a configuration item, the CM system might record a description of the change, accountability for the change, identification of all configuration items affected, status (e.g. pending or completed), and date and time of the change. This might be recorded in an audit trail of changes made or change control records;

f) the approach to version control and unique referencing of TOE versions (e.g. covering the release of patches in operating systems, and the subsequent detection of their application).

ACM_CAP.3.8C

3:ACM_CAP.3-10 The evaluator ***shall check*** the CM documentation to ascertain that it includes the CM system records identified by the CM plan.

942 The output produced by the CM system should provide the evidence that the evaluator needs to be confident that the CM plan is being applied, and also that all configuration items are being maintained by the CM system as required by ACM_CAP.3.9C. Example output could include change control forms, or configuration item access approval forms.

3:ACM_CAP.3-11 The evaluator ***shall examine*** the evidence to determine that the CM system is being used as it is described in the CM plan.

943 The evaluator should select and examine a sample of evidence covering each type of CM-relevant operation that has been performed on a configuration item (e.g. creation, modification, deletion, reversion to an earlier version) to confirm that all operations of the CM system have been carried out in line with documented procedures. The evaluator confirms that the evidence includes all the information

identified for that operation in the CM plan. Examination of the evidence may require access to a CM tool that is used. The evaluator may choose to sample the evidence.

944 For guidance on sampling see Annex B.2.

945 Further confidence in the correct operation of the CM system and the effective maintenance of configuration items may be established by means of interview with selected development staff. In conducting such interviews, the evaluator should aim to gain a deeper understanding of how the CM system is used in practice as well as to confirm that the CM procedures are being applied as described in the CM documentation. Note that such interviews should complement rather than replace the examination of documentary evidence, and may not be necessary if the documentary evidence alone satisfies the requirement. However, given the wide scope of the CM plan it is possible that some aspects (e.g. roles and responsibilities) may not be clear from the CM plan and records alone. This is one case where clarification may be necessary through interviews.

946 It is expected that the evaluator will visit the development site in support of this activity.

947 For guidance on site visits see Annex B.5.

ACM_CAP.3.9C

3:ACM_CAP.3-12 The evaluator ***shall check*** that the configuration items identified in the configuration list are being maintained by the CM system.

948 The CM system employed by the developer should maintain the integrity of the TOE. The evaluator should check that for each type of configuration item (e.g. high-level design or source code modules) contained in the configuration list there are examples of the evidence generated by the procedures described in the CM plan. In this case, the approach to sampling will depend upon the level of granularity used in the CM system to control CM items. Where, for example, 10,000 source code modules are identified in the configuration list, a different sampling strategy should be applied compared to the case in which there are only 5, or even 1. The emphasis of this activity should be on ensuring that the CM system is being operated correctly, rather than on the detection of any minor error.

949 For guidance on sampling see Annex B.2.

950 ACM_CAP.3.10C

3:ACM_CAP.3-13 The evaluator ***shall examine*** the CM access control measures described in the CM plan to determine that they are effective in preventing unauthorised access to the configuration items.

951 The evaluator may use a number of methods to determine that the CM access control measures are effective. For example, the evaluator may exercise the access control measures to ensure that the procedures could not be bypassed. The

evaluator may use the outputs generated by the CM system procedures and already examined as part of the work unit 3:ACM_CAP.3-12. The evaluator may also witness a demonstration of the CM system to ensure that the access control measures employed are operating effectively.

7.4.2 Evaluation of CM scope (ACM_SCP.1)

7.4.2.1 Objectives

Interp Note : The following paragraph is changed as a result of Interpretation 038.

952 The objective of this sub-activity is to determine whether ~~as a minimum~~ the developer performs configuration management on the TOE implementation representation, design, tests, user and administrator guidance, and the CM documentation.

7.4.2.2 Input

Interp Note : The following paragraph is changed as a result of Interpretation 004.

953 The evaluation evidence for this sub-activity is:

a) the configuration ~~management documentation~~ item list.

7.4.2.3 Evaluator action

954 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_SCP.1.1E.

7.4.2.3.1 Action ACM_SCP.1.1E

ACM_SCP.1.1C

Interp Note : The following element is changed as a result of Interpretation 004.

3:ACM_SCP.1-1 The evaluator ***shall check*** that the configuration item list includes the ~~minimum~~ set of items required by the CC ~~to be tracked by the CM system~~.

Interp Note : The following paragraph is changed as a result of Interpretations 004 and 038.

955 The list should include at least the following ~~as a minimum~~:

~~a) all documentation required to meet the target level of assurance;~~

~~b) other design documentation (e.g. low-level design);~~

~~c) test software (if applicable);~~

~~d) the TOE implementation representation (i.e. the components or subsystems that compose the TOE). For a software-only TOE, the implementation representation may consist solely of source code; for a TOE that includes a hardware platform, the implementation representation may refer to a combination of software, firmware and a description of the hardware (or a reference platform).~~

a) the TOE implementation representation (i.e. the components or subsystems that compose the TOE). For a software-only TOE, the implementation representation may consist solely of source code; for a TOE that includes a hardware platform, the implementation representation may refer to a combination of software, firmware and a description of the hardware;

b) the evaluation evidence documentation required to by the assurance components in the ST.

Interp Note : The following work unit is deleted as a result of Interpretation 004.

~~ACM_SCP.1.2C~~

3:ACM_SCP.1-2 ~~The evaluator ***shall examine*** the CM documentation to determine that the procedures describe how the status of each configuration item can be tracked throughout the lifecycle of the TOE.~~

956 ~~The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:~~

~~a) how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;~~

~~b) how configuration items are assigned unique identifiers and how they are entered into the CM system;~~

~~c) the method to be used to identify superseded versions of a configuration item;~~

~~d) the method to be used for identifying and tracking configuration items through each stage of the TOE development and maintenance lifecycle (i.e. requirements specification, design, source code development, through to object code generation and on to executable code, module testing, implementation and operation);~~

~~e) the method used for assigning the current status of the configuration item at a given point in time and for tracking each configuration item through the various levels of representation at the development phase (i.e. source code development, through to object code generation and on to executable code, module testing and documentation);~~

~~f) the method used for identifying correspondence between configuration items such that if one configuration item is changed it can be determined which other configuration items will also need to be changed.~~

957 ~~The analysis of the CM documentation for some of this information may have been satisfied by work units detailed under ACM_CAP.~~

7.5 Delivery and operation activity

958 The purpose of the delivery and operation activity is to judge the adequacy of the documentation of the procedures used to ensure that the TOE is installed, generated, and started in the same way the developer intended it to be and that it is delivered without modification. This includes both the procedures taken while the TOE is in transit, as well as the initialisation, generation, and start-up procedures.

959 The delivery and operation activity at EAL3 contains sub-activities related to the following components:

a) ADO_DEL.1;

b) ADO_IGS.1.

7.5.1 Evaluation of delivery (ADO_DEL.1)

7.5.1.1 Objectives

Interp Note : The following paragraph is changed as a result of Interpretation 016.

960 The objective of this sub-activity is to determine whether the delivery documentation describes all procedures used to maintain ~~integrity~~ security of the TOE when distributing the TOE to the user's site.

7.5.1.2 Input

961 The evaluation evidence for this sub-activity is:

a) the delivery documentation.

7.5.1.3 Evaluator actions

962 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_DEL.1.1E;

b) implied evaluator action based on ADO_DEL.1.2D.

7.5.1.3.1 Action ADO_DEL.1.1E

ADO_DEL.1.1C

3:ADO_DEL.1-1 The evaluator ***shall examine*** the delivery documentation to determine that it describes all procedures that are necessary to maintain security when distributing versions of the TOE or parts of it to the user's site.

963 Interpretation of the term *necessary* will need to consider the nature of the TOE and information contained in the ST. The level of protection provided should be commensurate with the assumptions, threats, organisational security policies, and

security objectives identified in the ST. In some cases these may not be explicitly expressed in relation to delivery. The evaluator should determine that a balanced approach has been taken, such that delivery does not present an obvious weak point in an otherwise secure development process.

Interp Note : The following two paragraphs are changed as a result of Interpretation 016.

964 The delivery procedures describe proper procedures to determine the identification of the TOE and to maintain ~~integrity~~ security of the TOE during transfer of the TOE or its component parts. The procedures describe which parts of the TOE need to be covered by these procedures. It should contain procedures for physical or electronic (e.g. for downloading off the Internet) distribution where applicable. The delivery procedures refer to the entire TOE, including applicable software, hardware, firmware and documentation.

965 ~~The emphasis on integrity is not surprising, since integrity will always be of concern for TOE delivery. Where confidentiality and availability are of concern, they also should be considered under this work unit.~~ The emphasis in the delivery documentation is likely to be on measures related to integrity, as technical measures are required to be applied to maintain integrity during the TOE delivery. However, confidentiality and availability of the delivery will be of concern in the delivery of some TOEs; procedures relating to these aspects of the secure delivery should also be discussed in the procedures.

966 The delivery procedures should be applicable across all phases of delivery from the production environment to the installation environment (e.g. packaging, storage and distribution).

Interp Note : The following two paragraphs are added as a result of Interpretation 116.

Standard commercial practice for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution, the public mail or a private distribution service may be acceptable.

The suitability of the choice of the delivery procedures is influenced by the TOE (e.g. whether it is software or hardware) and by the security objectives. In cases where the delivery procedures differ for different parts of the TOE, the totality of procedures are suitable to meet the overall security objectives.

Interp Note : The following work unit and its two guidance are deleted as a result of Interpretation 116.

3:ADO_DEL.1-2 ~~The evaluator ***shall examine*** the delivery procedures to determine that the chosen procedure and the part of the TOE it covers is suitable to meet the security objectives.~~

967 ~~The suitability of the choice of the delivery procedures is influenced by the specific TOE (e.g. whether it is software or hardware) and by the security objectives.~~

968 ~~Standard commercial practice for packaging and delivery may be acceptable This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution the public mail or a private distribution service may be acceptable.~~

7.5.1.3.2 Implied evaluator action

ADO_DEL.1.2D

3:ADO_DEL.1-3 The evaluator ***shall examine*** aspects of the delivery process to determine that the delivery procedures are used.

969 The approach taken by the evaluator to check the application of delivery procedures will depend on the nature of the TOE, and the delivery process itself. In addition to examination of the procedures themselves, the evaluator should seek some assurance that they are applied in practice. Some possible approaches are:

a) a visit to the distribution site(s) where practical application of the procedures may be observed;

b) examination of the TOE at some stage during delivery, or at the user's site (e.g. checking for tamper proof seals);

c) observing that the process is applied in practice when the evaluator obtains the TOE through regular channels;

d) questioning end users as to how the TOE was delivered.

970 For guidance on site visits see Annex B.5.

971 It may be the case of a newly developed TOE that the delivery procedures have yet to be exercised. In these cases, the evaluator has to be satisfied that appropriate procedures and facilities are in place for future deliveries and that all personnel involved are aware of their responsibilities. The evaluator may request a "dry run" of a delivery if this is practical. If the developer has produced other similar products, then an examination of procedures in their use may be useful in providing assurance.

7.5.2 Evaluation of installation, generation and start-up (ADO_IGS.1)

7.5.2.1 Objectives

972 The objective of this sub-activity is to determine whether the procedures and steps for the secure installation, generation, and start-up of the TOE have been documented and result in a secure configuration.

7.5.2.2 Input

973 The evaluation evidence for this sub-activity is:

a) the administrator guidance;

b) the secure installation, generation, and start-up procedures;

c) the TOE suitable for testing.

7.5.2.3 Application notes

974 The installation, generation, and start-up procedures refer to all installation, generation, and start-up procedures, regardless of whether they are performed at the user's site or at the development site that are necessary to progress the TOE to the secure configuration as described in the ST.

7.5.2.4 Evaluator actions

975 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_IGS.1.1E;

b) ADO_IGS.1.2E.

7.5.2.4.1 Action ADO_IGS.1.1E

ADO_IGS.1.1C

3:ADO_IGS.1-1 The evaluator ***shall check*** that the procedures necessary for the secure installation, generation and start-up of the TOE have been provided.

976 If it is not anticipated that the installation, generation, and start-up procedures will or can be reapplied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

7.5.2.4.2 Action ADO_IGS.1.2E

3:ADO_IGS.1-2 The evaluator ***shall examine*** the provided installation, generation, and start-up procedures to determine that they describe the steps necessary for secure installation, generation, and start-up of the TOE.

977 If it is not anticipated that the installation, generation, and start-up procedures will or can be reapplied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

978 The installation, generation, and start-up procedures may provide detailed information about the following:

a) changing the installation specific security characteristics of entities under the control of the TSF;

b) handling exceptions and problems;

c) minimum system requirements for secure installation if applicable.

979 In order to confirm that the installation, generation, and start-up procedures result in a secure configuration, the evaluator may follow the developer's procedures and may perform the activities that customers are usually expected to perform to install, generate, and start-up the TOE (if applicable to the TOE), using the supplied guidance documentation only. This work unit might be performed in conjunction with the 3:ATE_IND.2-2 work unit.

7.6 Development activity

980 The purpose of the development activity is to assess the design documentation in terms of its adequacy to understand how the TSF provides the security functions of the TOE. This understanding is achieved through examination of increasingly refined descriptions of the TSF design documentation. Design documentation consists of a functional specification (which describes the external interfaces of the TOE) and a high-level design (which describes the architecture of the TOE in terms of internal subsystems). There is also a representation correspondence (which maps representations of the TOE to one another in order to ensure consistency).

981 The development activity at EAL3 contains sub-activities related to the following components:

a) ADV_FSP.1;

b) ADV_HLD.2;

c) ADV_RCR.1.

7.6.1 Application notes

982 The CC requirements for design documentation are levelled by formality. The CC considers a document's degree of formality (that is, whether it is informal, semiformal or formal) to be hierarchical. An informal document is one that is expressed in a natural language. The methodology does not dictate the specific language that must be used; that issue is left for the scheme. The following paragraphs differentiate the contents of the different informal documents.

983 An informal functional specification comprises a description the security functions (at a level similar to that of the TOE summary specification) and a description of the externally-visible interfaces to the TSF. For example, if an operating system presents the user with a means of self-identification, of creating files, of modifying or deleting files, of setting permissions defining what other users may access files, and of communicating with remote machines, its functional specification would contain descriptions of each of these functions. If there are also audit functions that detect and record the occurrences of such events, descriptions of these audit functions would also be expected to be part of the functional specification; while these functions are technically not directly invoked by the user at the external interface, they certainly are affected by what occurs at the user's external interface.

984 An informal high-level design is expressed in terms of sequences of actions that occur in each subsystem in response to stimulus at its interface. For example, a firewall might be composed of subsystems that deal with packet filtering, with remote administration, with auditing, and with connection-level filtering. The high-level design description of the firewall would describe the actions that are taken, in terms of what actions each subsystem takes when an incoming packet arrives at the firewall.

985 Informality of the demonstration of correspondence need not be in a prose form; a simple two-dimensional mapping may be sufficient. For example, a matrix with modules listed along one axis and subsystems listed along the other, with the cells identifying the correspondence of the two, would serve to provide an adequate informal correspondence between the high-level design and the low-level design.

7.6.2 Evaluation of functional specification (ADV_FSP.1)

7.6.2.1 Objectives

986 The objective of this sub-activity is to determine whether the developer has provided an adequate description of the security functions of the TOE and whether the security functions provided by the TOE are sufficient to satisfy the security functional requirements of the ST.

7.6.2.2 Input

987 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance.

7.6.2.3 Evaluator actions

988 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_FSP.1.1E;

b) ADV_FSP.1.2E.

7.6.2.3.1 Action ADV_FSP.1.1E

ADV_FSP.1.1C

3:ADV_FSP.1-1 The evaluator ***shall examine*** the functional specification to determine that it contains all necessary informal explanatory text.

989 If the entire functional specification is informal, this work unit is not applicable and is therefore considered to be satisfied.

990 Supporting narrative descriptions are necessary for those portions of the functional specification that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_FSP.1.2C

3:ADV_FSP.1-2 The evaluator ***shall examine*** the functional specification to determine that it is internally consistent.

991 The evaluator validates the functional specification by ensuring that the descriptions of the interfaces making up the TSFI are consistent with the descriptions of the functions of the TSF.

992 For guidance on consistency analysis see Annex B.3.

ADV_FSP.1.3C

3:ADV_FSP.1-3 The evaluator ***shall examine*** the functional specification to determine that it identifies all of the external TOE security function interfaces.

993 The term *external* refers to that which is visible to the user. External interfaces to the TOE are either direct interfaces to the TSF or interfaces to non-TSF portions of the TOE. However, these non-TSF interfaces might have eventual access to the TSF. These external interfaces that directly or indirectly access the TSF collectively make up the TOE security function interface (TSFI). Figure 7.1 shows a TOE with TSF (shaded) portions and non-TSF (empty) portions. This TOE has three external interfaces: interface *c* is a direct interface to the TSF; interface *b* is an indirect interface to the TSF; and interface *a* is an interface to non-TSF portions of the TOE. Therefore, interfaces *b* and *c* make up the TFSI.

994 It should be noted that all security functions reflected in the functional requirements of CC Part 2 (or in extended components thereof) will have some sort of externally-visible manifestation. While not all of these are necessarily interfaces from which the security function can be tested, they are all externally-visible to some extent and must therefore be included in the functional specification.

995 For guidance on determining the TOE boundary see Annex B.6.

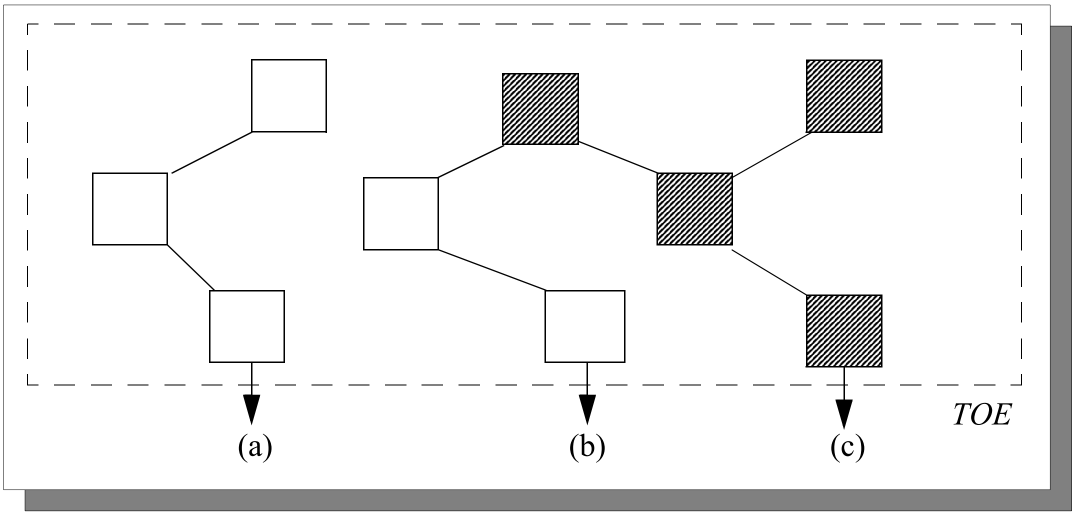

Figure 7.1 TSF Interfaces

1:ADV_FSP.1-4 The evaluator ***shall examine*** the functional specification to determine that it describes all of the external TOE security function interfaces.

996 For a TOE that has no threat of malicious users (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are rightfully excluded from its ST), the only interfaces that are described in the functional specification (and expanded upon in the other TSF representation descriptions) are those to and from the TSF. The absence of FPT_PHP, FPT_RVM, and FPT_SEP presumes there is no concern for any sort of bypassing of the security features; therefore, there is no concern with any possible impact that other interfaces might have on the TSF.

997 On the other hand, if the TOE has a threat of malicious users or bypass (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are included in its ST), all external interfaces are described in the functional specification, but only to the extent that the effect of each is made clear: interfaces to the security functions (i.e. interfaces *b* and *c* in Figure 7.1) are completely described, while other interfaces are described only to the extent that it is clear that the TSF is inaccessible through the interface (i.e. that the interface is of type *a*, rather than *b* in Figure 7.1). The inclusion of FPT_PHP, FPT_RVM, and FPT_SEP implies a concern that all interfaces might have some effect upon the TSF. Because each external interface is a potential TSF interface, the functional specification must contain a description of each interface in sufficient detail so that an evaluator can determine whether the interface is security relevant.

998 Some architectures lend themselves to readily provide this interface description in sufficient detail for groups of external interfaces. For example, a kernel architecture is such that all calls to the operating system are handled by kernel

programs; any calls that might violate the TSP must be called by a program with the privilege to do so. All programs that execute with privilege must be included in the functional specification. Any program external to the kernel that executes without privilege is incapable of affecting the TSP (i.e. such programs are interfaces of type *a*, rather than *b* in Figure 7.1) and may, therefore, be excluded from the functional specification. It is worth noting that, while the evaluator's understanding of the interface description can be expedited in cases where there is a kernel architecture, such an architecture is not necessary.

3:ADV_FSP.1-5 The evaluator ***shall examine*** the presentation of the TSFI to determine that it adequately and correctly describes the behaviour of the TOE at each external interface describing effects, exceptions and error messages.

999 In order to assess the adequacy and correctness of an interface's presentation, the evaluator uses the functional specification, the TOE summary specification of the ST, and the user and administrator guidance to assess the following factors:

a) All security relevant user input parameters (or a characterisation of those parameters) should be identified. For completeness, parameters outside of direct user control should be identified if they are usable by administrators.

b) All security relevant behaviour described in the reviewed guidance should be reflected in the description of semantics in the functional specification. This should include an identification of the behaviour in terms of events and the effect of each event. For example, if an operating system provides a rich file system interface, where it provides a different error code for each reason why a file is not opened upon request (e.g. access denied, no such file, file is in use by another user, user is not authorised to open the file after 5pm, etc.), the functional specification should explain that a file is either opened upon request, or else that an error code is returned. (While the functional specification may enumerate all these different reasons for errors, it need not provide such detail.) The description of the semantics should include how the security requirements apply to the interface (e.g. whether the use of the interface is an auditable event and, if so, the information that can be recorded).

c) All interfaces are described for all possible modes of operation. If the TSF provides the notion of privilege, the description of the interface should explain how the interface behaves in the presence or absence of privilege.

d) The information contained in the descriptions of the security relevant parameters and syntax of the interface should be consistent across all documentation.

1000 Verification of the above is done by reviewing the functional specification and the TOE summary specification of the ST, as well as the user and administrator guidance provided by the developer. For example, if the TOE were an operating system and its underlying hardware, the evaluator would look for discussions of user-accessible programs, descriptions of protocols used to direct the activities of programs, descriptions of user-accessible databases used to direct the activities of programs, and for user interfaces (e.g. commands, application program interfaces) as applicable to the TOE under evaluation; the evaluator would also ensure that the processor instruction set is described.

1001 This review might be iterative, such that the evaluator would not discover the functional specification to be incomplete until the design, source code, or other evidence is examined and found to contain parameters or error messages that have been omitted from the functional specification.

ADV_FSP.1.4C

3:ADV_FSP.1-6 The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

1002 In order to assess the completeness of the TSF representation, the evaluator consults the TOE summary specification of the ST, the user guidance, and the administrator guidance. None of these should describe security functions that are absent from the TSF presentation of the functional specification.

7.6.2.3.2 Action ADV_FSP.1.2E

3:ADV_FSP.1-7 The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the TOE security functional requirements.

1003 To ensure that all ST security functional requirements are covered by the functional specification, the evaluator may construct a map between the TOE summary specification and the functional specification. Such a map might be already provided by the developer as evidence for meeting the correspondence (ADV_RCR.*) requirements, in which case the evaluator need only verify the completeness of this mapping, ensuring that all security functional requirements are mapped onto applicable TSFI presentations in the functional specification.

3:ADV_FSP.1-8 The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the TOE security functional requirements.

1004 For each interface to a security function with specific characteristics, the detailed information in the functional specification must be exactly as it is specified in the ST. For example, if the ST contains user authentication requirements that the password length must be eight characters, the TOE must have eight-character passwords; if the functional specification describes six-character fixed length passwords, the functional specification would not be an accurate instantiation of the requirements.

1005 For each interface in the functional specification that operates on a controlled resource, the evaluator determines whether it returns an error code that indicates a possible failure due to enforcement of one of the security requirements; if no error code is returned, the evaluator determines whether an error code should be returned. For example, an operating system might present an interface to OPEN a controlled object. The description of this interface may include an error code that indicates that access was not authorised to the object. If such an error code does not exist, the evaluator should confirm that this is appropriate (because, perhaps, access mediation is performed on READs and WRITEs, rather than on OPENs).

7.6.3 Evaluation of high-level design (ADV_HLD.2)

7.6.3.1 Objectives

1006 The objective of this sub-activity is to determine whether the high-level design provides a description of the TSF in terms of major structural units (i.e. subsystems), provides a description of the interfaces to these structural units, and is a correct realisation of the functional specification.

7.6.3.2 Input

1007 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design.

7.6.3.3 Evaluator actions

1008 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_HLD.2.1E;

b) ADV_HLD.2.2E.

7.6.3.3.1 Action ADV_HLD.2.1E

ADV_HLD.2.1C

3:ADV_HLD.2-1 The evaluator ***shall examine*** the high-level design to determine that it contains all necessary informal explanatory text.

1009 If the entire high-level design is informal, this work unit is not applicable and is therefore considered to be satisfied.

1010 Supporting narrative descriptions are necessary for those portions of the high-level design that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_HLD.2.2C

3:ADV_HLD.2-2 The evaluator ***shall examine*** the presentation of the high-level design to determine that it is internally consistent.

1011 For guidance on consistency analysis see Annex B.3.

1012 The evaluator validates the subsystem interface specifications by ensuring that the interface specifications are consistent with the description of the purpose of the subsystem.

ADV_HLD.2.3C

3:ADV_HLD.2-3 The evaluator ***shall examine*** the high-level design to determine that the TSF is described in terms of subsystems.

1013 With respect to the high-level design, the term *subsystem* refers to large, related units (such as memory-management, file-management, process-management). Breaking a design into the basic functional areas aids in the understanding of the design.

1014 The primary purpose for examining the high-level design is to aid the evaluator's understanding of the TOE. The developer's choice of subsystem definition, and of the grouping of TSFs within each subsystem, are an important aspect of making the high-level design useful in understanding the TOE's intended operation. As part of this work unit, the evaluator should make an assessment as to the appropriateness of the number of subsystems presented by the developer, and also of the choice of grouping of functions within subsystems. The evaluator should ensure that the decomposition of the TSF into subsystems is sufficient for the evaluator to gain a high-level understanding of how the functionality of the TSF is provided.

1015 The subsystems used to describe the high-level design need not be called "subsystems", but should represent a similar level of decomposition. For example, the design may be decomposed using "layers" or "managers".

1016 There may be some interaction between the choice of subsystem definition and the scope of the evaluator's analysis. A discussion on this interaction is found following work unit 3:ADV_HLD.2-10.

ADV_HLD.2.4C

3:ADV_HLD.2-4 The evaluator ***shall examine*** the high-level design to determine that it describes the security functionality of each subsystem.

1017 The security functional behaviour of a subsystem is a description of what the subsystem does. This should include a description of any actions that the subsystem may be directed to perform through its functions and the effects the subsystem may have on the security state of the TOE (e.g. changes in subjects, objects, security databases).

ADV_HLD.2.5C

3:ADV_HLD.2-5 The evaluator ***shall check*** the high-level design to determine that it identifies all hardware, firmware, and software required by the TSF.

1018 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

1019 If the ST contains the optional statement of security requirements for the IT environment, the evaluator compares the list of hardware, firmware, or software required by the TSF as stated in the high-level design to the statement of security requirements for the IT environment to determine that they agree. The information in the ST characterises the underlying abstract machine on which the TOE will execute.

1020 If the high-level design includes security requirements for the IT environment that are not included in the ST, or if they differ from those included in the ST, this inconsistency is assessed by the evaluator under Action ADV_HLD.2.2E.

3:ADV_HLD.2-6 The evaluator ***shall examine*** the high-level design to determine that it includes a presentation of the functions provided by the supporting protection mechanisms implemented in the underlying hardware, firmware, or software.

1021 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

1022 The presentation of the functions provided by the underlying abstract machine on which the TOE executes need not be at the same level of detail as the presentation of functions that are part of the TSF. The presentation should explain how the TOE uses the functions provided in the hardware, firmware, or software that implement the security requirements for the IT environment that the TOE is dependent upon to support the TOE security objectives.

1023 The statement of security requirements for the IT environment may be abstract, particularly if it is intended to be capable of being satisfied by a variety of different combinations of hardware, firmware, or software. As part of the Tests activity, where the evaluator is provided with at least one instance of an underlying machine that is claimed to satisfy the security requirements for the IT environment, the evaluator can determine whether it provides the necessary security functions for the TOE. This determination by the evaluator does not require testing or analysis of the underlying machine; it is only a determination that the functions expected to be provided by it actually exist.

ADV_HLD.2.6C

3:ADV_HLD.2-7 The evaluator ***shall check*** that the high-level design identifies the interfaces to the TSF subsystems.

1024 The high-level design includes, for each subsystem, the name of each of its entry points.

ADV_HLD.2.7C

3:ADV_HLD.2-8 The evaluator ***shall check*** that the high-level design identifies which of the interfaces to the subsystems of the TSF are externally visible.

1025 As discussed under work unit 3:ADV_FSP.1-3, external interfaces (i.e. those visible to the user) may directly or indirectly access the TSF. Any external interface that accesses the TSF either directly or indirectly is included in the identification for this work unit. External interfaces that do not access the TSF need not be included.

ADV_HLD.2.8C

3:ADV_HLD.2-9 The evaluator ***shall examine*** the high-level design to determine that it describes the interfaces to each subsystem in terms of their purpose and method of use, and provides details of effects, exceptions and error messages, as appropriate.

1026 The high-level design should include descriptions in terms of the purpose and method of use for all interfaces of each subsystem. Such descriptions may be provided in general terms for some interfaces, and in more detail for others. In determining the level of detail of effects, exceptions and error messages that should be provided, the evaluator should consider the purposes of this analysis and the uses made of the interface by the TOE. For example, the evaluator needs to understand the nature of the interactions between subsystems to establish confidence that the TOE design is sound, and may be able to obtain this understanding with only a general description of some of the interfaces between subsystems. In particular, internal subsystem entry points that are not called by any other subsystem would not normally require detailed descriptions.

1027 The level of detail may also depend on the testing approach adopted to meet the ATE_DPT requirement. For example, a different amount of detail may be needed for a testing approach that tests only through external interfaces than one that tests through both external and internal subsystem interfaces.

1028 Detailed descriptions would include details of any input and output parameters, of the effects of the interface, and of any exceptions or error messages it produces. In the case of external interfaces, the required description is probably included in the functional specification and can be referenced in the high-level design without replication.

ADV_HLD.2.9C

3:ADV_HLD.2-10 The evaluator ***shall check*** that the high-level design describes the separation of the TOE into TSP-enforcing and other subsystems.

1029 The TSF comprises all the parts of the TOE that have to be relied upon for enforcement of the TSP. Because the TSF includes both functions that directly enforce the TSP, and also those functions that, while not directly enforcing the TSP, contribute to the enforcement of the TSP in a more indirect manner, all TSP-enforcing subsystems are contained in the TSF. Subsystems that play no role in TSP enforcement are not part of the TSF. An entire subsystem is part of the TSF if any portion of it is.

1030 As explained under work unit 3:ADV_HLD.2-3, the developer's choice of subsystem definition, and of the grouping of TSFs within each subsystem, are important aspects of making the high-level design useful in understanding the TOE's intended operation. However, the choice of grouping of TSFs within subsystems also affects the scope of the TSF, because a subsystem with *any* function that directly or indirectly enforces the TSP is part of the TSF. While the goal of understandability is important, it is also helpful to limit the extent of the TSF so as to reduce the amount of analysis that is required. The two goals of understandability and scope reduction may sometimes work against each other. The evaluator should bear this in mind when assessing the choice of subsystem definition.

7.6.3.3.2 Action ADV_HLD.2.2E

3:ADV_HLD.2-11 The evaluator ***shall examine*** the high-level design to determine that it is an accurate instantiation of the TOE security functional requirements.

1031 The evaluator analyses the high-level design for each TOE security function to ensure that the function is accurately described. The evaluator also ensures that the function has no dependencies that are not included in the high-level design.

1032 The evaluator also analyses the security requirements for the IT environment in both the ST and the high-level design to ensure that they agree. For example, if the ST includes TOE security functional requirements for the storage of an audit trail, and the high-level design stated that audit trail storage is provided by the IT environment, then the high-level design is not an accurate instantiation of the TOE security functional requirements.

1033 The evaluator should validate the subsystem interface specifications by ensuring that the interface specifications are consistent with the description of the purpose of the subsystem.

3:ADV_HLD.2-12 The evaluator ***shall examine*** the high-level design to determine that it is a complete instantiation of the TOE security functional requirements.

1034 To ensure that all ST security functional requirements are covered by the high-level design, the evaluator may construct a map between the TOE security functional requirements and the high-level design.

7.6.4 Evaluation of representation correspondence (ADV_RCR.1)

7.6.4.1 Objectives

1035 The objective of this sub-activity is to determine whether the developer has correctly and completely implemented the requirements of the ST and functional specification in the high-level design.

7.6.4.2 Input

1036 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the correspondence analysis between the TOE summary specification and the functional specification;

e) the correspondence analysis between the functional specification and the high-level design.

7.6.4.3 Evaluator actions

1037 This sub-activity comprises one CC Part 3 evaluator action element:

a) ADV_RCR.1.1E.

7.6.4.3.1 Action ADV_RCR.1.1E

ADV_RCR.1.1C

3:ADV_RCR.1-1 The evaluator ***shall examine*** the correspondence analysis between the TOE summary specification and the functional specification to determine that the functional specification is a correct and complete representation of the TOE security functions.

1038 The evaluator's goal in this work unit is to determine that all security functions identified in the TOE summary specification are represented in the functional specification and that they are represented accurately.

1039 The evaluator reviews the correspondence between the TOE security functions of the TOE summary specification and the functional specification. The evaluator looks for consistency and accuracy in the correspondence. Where the correspondence analysis indicates a relationship between a security function of the TOE summary specification and an interface description in the functional specification, the evaluator verifies that the security functionality of both are the same. If the security functions of the TOE summary specification are correctly and completely present in the corresponding interface, this work unit will be satisfied.

1040 This work unit may be done in conjunction with work units 3:ADV_FSP.1-7 and 3:ADV_FSP.1-8.

3:ADV_RCR.1-2 The evaluator ***shall examine*** the correspondence analysis between the functional specification and the high-level design to determine that the high-level design is a correct and complete representation of the functional specification.

1041 The evaluator uses the correspondence analysis, the functional specification, and the high-level design to ensure that it is possible to map each security function identified in the functional specification onto a TSF subsystem described in the high-level design. For each security function, the correspondence indicates which TSF subsystems are involved in the support of the function. The evaluator verifies that the high-level design includes a description of a correct realisation of each security function.

7.7 Guidance documents activity

1042 The purpose of the guidance document activity is to judge the adequacy of the documentation describing how to use the operational TOE. Such documentation includes both that aimed at trusted administrators and non-administrator users whose incorrect actions could adversely affect the security of the TOE, as well as that aimed at untrusted users whose incorrect actions could adversely affect the security of their own data.

1043 The guidance documents activity at EAL3 contains sub-activities related to the following components:

a) AGD_ADM.1;

b) AGD_USR.1.

7.7.1 Application notes

1044 The guidance documents activity applies to those functions and interfaces which are related to the security of the TOE. The secure configuration of the TOE is described in the ST.

7.7.2 Evaluation of administrator guidance (AGD_ADM.1)

7.7.2.1 Objectives

1045 The objective of this sub-activity is to determine whether the administrator guidance describes how to administer the TOE in a secure manner.

7.7.2.2 Application notes

1046 The term *administrator* is used to indicate a human user who is trusted to perform security critical operations within the TOE, such as setting TOE configuration parameters. The operations may affect the enforcement of the TSP, and the administrator therefore possesses specific privileges necessary to perform those operations. The role of the administrator(s) has to be clearly distinguished from the role of non-administrative users of the TOE.

1047 There may be different administrator roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF such as auditor, administrator, or daily-management. Each role can encompass an extensive set of capabilities, or can be a single one. The capabilities of these roles and their associated privileges are described in the FMT class. Different administrator roles and groups should be taken into consideration by the administrator guidance.

7.7.2.3 Input

1048 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures.

7.7.2.4 Evaluator actions

1049 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_ADM.1.1E.

7.7.2.4.1 Action AGD_ADM.1.1E

AGD_ADM.1.1C

3:AGD_ADM.1-1 The evaluator ***shall examine*** the administrator guidance to determine that it describes the administrative security functions and interfaces available to the administrator of the TOE.

1050 The administrator guidance should contain an overview of the security functionality that is visible at the administrator interfaces.

1051 The administrator guidance should identify and describe the purpose, behaviour, and interrelationships of the administrator security interfaces and functions.

1052 For each administrator security interface and function, the administrator guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system calls, menu selection, command button);

b) describe the parameters to be set by the administrator, their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_ADM.1.2C

3:AGD_ADM.1-2 The evaluator ***shall examine*** the administrator guidance to determine that it describes how to administer the TOE in a secure manner.

1053 The administrator guidance describes how to operate the TOE according to the TSP in an IT environment that is consistent with the one described in the ST.

AGD_ADM.1.3C

3:AGD_ADM.1-3 The evaluator ***shall examine*** the administrator guidance to determine that it contains warnings about functions and privileges that should be controlled in a secure processing environment.

1054 The configuration of the TOE may allow users to have dissimilar privileges to make use of the different functions of the TOE. This means that some users may be authorised to perform certain functions while other users may not be so authorised. These functions and privileges should be described by the administrator guidance.

1055 The administrator guidance identifies the functions and privileges that must be controlled, the types of controls required for them, and the reasons for such controls. Warnings address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_ADM.1.4C

3:AGD_ADM.1-4 The evaluator ***shall examine*** the administrator guidance to determine that it describes all assumptions regarding user behaviour that are relevant to the secure operation of the TOE.

1056 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the information that is of concern to the secure operation of the TOE need be included in the administrator guidance.

1057 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

AGD_ADM.1.5C

3:AGD_ADM.1-5 The evaluator ***shall examine*** the administrator guidance to determine that it describes all security parameters under the control of the administrator indicating secure values as appropriate.

1058 For each security parameter, the administrator guidance should describe the purpose of the parameter, the valid and default values of the parameter, and secure and insecure use settings of such parameters, both individually or in combination.

AGD_ADM.1.6C

3:AGD_ADM.1-6 The evaluator ***shall examine*** the administrator guidance to determine that it describes each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

1059 All types of security-relevant events are detailed, such that an administrator knows what events may occur and what action (if any) the administrator may have to take in order to maintain security. Security-relevant events that may occur during operation of the TOE (e.g. audit trail overflow, system crash, updates to user records, such as when a user account is removed when the user leaves the organisation) are adequately defined to allow administrator intervention to maintain secure operation.

AGD_ADM.1.7C

3:AGD_ADM.1-7 The evaluator ***shall examine*** the administrator guidance to determine that it is consistent with all other documents supplied for evaluation.

1060 The ST in particular may contain detailed information on any warnings to the TOE administrators with regard to the TOE security environment and the security objectives.

1061 For guidance on consistency analysis see Annex B.3.

AGD_ADM.1.8C

3:AGD_ADM.1-8 The evaluator ***shall examine*** the administrator guidance to determine that it describes all IT security requirements for the IT environment of the TOE that are relevant to the administrator.

1062 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

1063 This work unit relates to IT security requirements only and not to any organisational security policies.

1064 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare them with the administrator guidance to ensure that all security requirements of the ST that are relevant to the administrator are described appropriately in the administrator guidance.

7.7.3 Evaluation of user guidance (AGD_USR.1)

7.7.3.1 Objectives

1065 The objectives of this sub-activity are to determine whether the user guidance describes the security functions and interfaces provided by the TSF and whether this guidance provides instructions and guidelines for the secure use of the TOE.

7.7.3.2 Application notes

1066 There may be different user roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF. The capabilities of these roles and their associated privileges are described in the FMT class. Different user roles and groups should be taken into consideration by the user guidance.

7.7.3.3 Input

1067 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures.

7.7.3.4 Evaluator actions

1068 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_USR.1.1E.

7.7.3.4.1 Action AGD_USR.1.1E

AGD_USR.1.1C

3:AGD_USR.1-1 The evaluator ***shall examine*** the user guidance to determine that it describes the security functions and interfaces available to the non-administrative users of the TOE.

1069 The user guidance should contain an overview of the security functionality that is visible at the user interfaces.

1070 The user guidance should identify and describe the purpose of the security interfaces and functions.

AGD_USR.1.2C

3:AGD_USR.1-2 The evaluator ***shall examine*** the user guidance to determine that it describes the use of user-accessible security functions provided by the TOE.

1071 The user guidance should identify and describe the behaviour and interrelationship of the security interfaces and functions available to the user.

1072 If the user is allowed to invoke a TOE security function, the user guidance provides a description of the interfaces available to the user for that function.

1073 For each interface and function, the user guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system call, menu selection, command button);

b) describe the parameters to be set by the user and their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_USR.1.3C

3:AGD_USR.1-3 The evaluator ***shall examine*** the user guidance to determine that it contains warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

1074 The configuration of the TOE may allow users to have dissimilar privileges in making use of the different functions of the TOE. This means that some users are authorised to perform certain functions, while other users may not be so authorised. These user-accessible functions and privileges are described by the user guidance.

1075 The user guidance should identify the functions and privileges that can be used, the types of commands required for them, and the reasons for such commands. The user guidance should contain warnings regarding the use of the functions and privileges that must be controlled. Warnings should address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_USR.1.4C

3:AGD_USR.1-4 The evaluator ***shall examine*** the user guidance to determine that it presents all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

1076 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the

information that is of concern to the secure operation of the TOE need be included in the user guidance.

1077 The user guidance should provide advice regarding effective use of the security functions (e.g. reviewing password composition practices, suggested frequency of user file backups, discussion on the effects of changing user access privileges).

1078 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

1079 The user guidance should indicate whether the user can invoke a function or whether the user requires the assistance of an administrator.

AGD_USR.1.5C

3:AGD_USR.1-5 The evaluator ***shall examine*** the user guidance to determine that it is consistent with all other documentation supplied for evaluation.

1080 The evaluator ensures that the user guidance and all other documents supplied for evaluation do not contradict each other. This is especially true if the ST contains detailed information on any warnings to the TOE users with regard to the TOE security environment and the security objectives.

1081 For guidance on consistency analysis see Annex B.3.

AGD_USR.1.6C

3:AGD_USR.1-6 The evaluator ***shall examine*** the user guidance to determine that it describes all security requirements for the IT environment of the TOE that are relevant to the user.

1082 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

1083 This work unit relates to IT security requirements only and not to any organisational security policies.

1084 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare that with the user guidance to ensure that all security requirements of the ST, that are relevant to the user, are described appropriately in the user guidance.

7.8 Life-cycle support activity

1085 The purpose of the life-cycle support activity is to determine the adequacy of the security procedures the developer uses during the development and maintenance of the TOE. Such procedures are intended to protect the TOE and its associated design information from interference or disclosure. Interference in the development process may allow the deliberate introduction of vulnerabilities. Disclosure of design information may allow vulnerabilities to be more easily exploited. The adequacy of the procedures will depend on the nature of the TOE and the development process.

1086 The life-cycle support activity at EAL3 contains a sub-activity related to the following component:

a) ALC_DVS.1.

7.8.1 Evaluation of development security (ALC_DVS.1)

7.8.1.1 Objectives

1087 The objective of this sub-activity is to determine whether the developer's security controls on the development environment are adequate to provide the confidentiality and integrity of the TOE design and implementation that is necessary to ensure that secure operation of the TOE is not compromised.

7.8.1.2 Input

1088 The evaluation evidence for this sub-activity is:

a) the ST;

b) the development security documentation.

1089 In addition, the evaluator may need to examine other deliverables to determine that the security controls are well-defined and followed. Specifically, the evaluator may need to examine the developer's configuration management documentation (the input for the ACM_CAP.3 and ACM_SCP.1 sub-activities). Evidence that the procedures are being applied is also required.

7.8.1.3 Evaluator actions

1090 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ALC_DVS.1.1E;

b) ALC_DVS.1.2E.

7.8.1.3.1 Action ALC_DVS.1.1E

ALC_DVS.1.1C

3:ALC_DVS.1-1 The evaluator ***shall examine*** the development security documentation to determine that it details all security measures used in the development environment that are necessary to protect the confidentiality and integrity of the TOE design and implementation.

1091 The evaluator determines what is necessary by first referring to the ST for any information that may assist in the determination of necessary protection, especially the sections on threats, organisational security policies and assumptions, although there may be no information provided explicitly. The statement of security objectives for the environment may also be useful in this respect.

1092 If no explicit information is available from the ST the evaluator will need to make a determination of the necessary measures, based upon a consideration of the intended environment for the TOE. In cases where the developer's measures are considered less than what is necessary, a clear justification should be provided for the assessment, based on a potential exploitable vulnerability.

1093 The following types of security measures are considered by the evaluator when examining the documentation:

a) *physical*, for example physical access controls used to prevent unauthorised access to the TOE development environment (during normal working hours and at other times);

b) *procedural*, for example covering:

- granting of access to the development environment or to specific parts of the environment such as development machines
- revocation of access rights when a person leaves the development team
- transfer of protected material out of the development environment
- admitting and escorting visitors to the development environment
- roles and responsibilities in ensuring the continued application of security measures, and the detection of security breaches.

c) *personnel*, for example any controls or checks made to establish the trustworthiness of new development staff;

d) *other security measures*, for example the logical protections on any development machines.

1094 The development security documentation should identify the locations at which development occurs, and describe the aspects of development performed, along with the security measures applied at each location. For example, development could occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites. Development includes such tasks as creating multiple copies of the TOE, where applicable. This work-unit should not overlap with those for ADO_DEL, but the evaluator should ensure that all aspects are covered by one sub-activity or the other.

1095 In addition, the development security documentation may describe different security measures that can be applied to different aspects of development in terms of their performance and the required inputs and outputs. For example, different procedures may be applicable to the development of different portions of the TOE, or to different stages of the development process.

3:ALC_DVS.1-2 The evaluator ***shall examine*** the development confidentiality and integrity policies in order to determine the sufficiency of the security measures employed.

1096 These include the policies governing:

a) what information relating to the TOE development needs to be kept confidential, and which members of the development staff are allowed to access such material;

b) what material must be protected from unauthorised modification in order to preserve the integrity of the TOE, and which members of the development staff are allowed to modify such material.

1097 The evaluator should determine that these policies are described in the development security documentation, that the security measures employed are consistent with the policies, and that they are complete.

1098 It should be noted that configuration management procedures will help protect the integrity of the TOE and the evaluator should avoid overlap with the work-units conducted for the ACM_CAP sub-activity. For example, the CM documentation may describe the security procedures necessary for controlling the roles or individuals who should have access to the development environment and who may modify the TOE.

1099 Whereas the ACM_CAP requirements are fixed, those for ALC_DVS, mandating only necessary measures, are dependent on the nature of the TOE, and on information that may be provided in the Security Environment section of the ST. For example, the ST may identify an organisational security policy that requires the TOE to be developed by staff who have security clearance. The evaluators would then determine that such a policy had been applied under this sub-activity.

ALC_DVS.1.2C

3:ALC_DVS.1-3 The evaluator ***shall check*** the development security documentation to determine that documentary evidence that would be produced as a result of application of the procedures has been generated.

1100 Where documentary evidence is produced the evaluator inspects it to ensure compliance with procedures. Examples of the evidence produced may include entry logs and audit trails. The evaluator may choose to sample the evidence.

1101 For guidance on sampling see Annex B.2.

7.8.1.3.2 Action ALC_DVS.1.2E

3:ALC_DVS.1-4 The evaluator ***shall examine*** the development security documentation and associated evidence to determine that the security measures are being applied.

1102 This work unit requires the evaluator to determine that the security measures described in the development security documentation are being followed, such that the integrity of the TOE and the confidentiality of associated documentation is being adequately protected. For example, this could be determined by examination of the documentary evidence provided. Documentary evidence should be supplemented by visiting the development environment. A visit to the development environment will allow the evaluator to:

a) observe the application of security measures (e.g. physical measures);

b) examine documentary evidence of application of procedures;

c) interview development staff to check awareness of the development security policies and procedures, and their responsibilities.

1103 A development site visit is a useful means of gaining confidence in the measures being used. Any decision not to make such a visit should be determined in consultation with the overseer.

1104 For guidance on site visits see Annex B.5.

7.9 Tests activity

1105 The purpose of this activity is to determine whether the TOE behaves as specified in the design documentation and in accordance with the TOE security functional requirements specified in the ST. This is accomplished by determining that the developer has tested the TSF against its functional specification and high-level design, gaining confidence in those test results by performing a sample of the developer's tests, and by independently testing a subset of the TSF.

1106 The tests activity at EAL3 contains sub-activities related to the following components:

a) ATE_COV.2;

b) ATE_DPT.1;

c) ATE_FUN.1;

d) ATE_IND.2.

7.9.1 Application notes

1107 The size and composition of the evaluator's test subset depends upon several factors discussed in the independent testing (ATE_IND.2) sub-activity. One such factor affecting the composition of the subset is *known public domain weaknesses,* information about which the evaluator needs access (e.g. from a scheme).

1108 The CC has separated coverage and depth from functional tests to increase the flexibility when applying the components of the families. However, the requirements of the families are intended to be applied together to confirm that the TSF operates according to its specification. This tight coupling of families has led to some duplication of evaluator work effort across sub-activities. These application notes are used to minimize duplication of text between sub-activities of the same activity and EAL.

7.9.1.1 Understanding the expected behaviour of the TOE

1109 Before the adequacy of test documentation can be accurately evaluated, or before new tests can be created, the evaluator has to understand the desired expected behaviour of a security function in the context of the requirements it is to satisfy.

1110 The evaluator may choose to focus on one security function of the TSF at a time. For each security function, the evaluator examines the ST requirement and the relevant parts of the functional specification, high-level design and guidance documentation to gain an understanding of the way the TOE is expected to behave.

1111 With an understanding of the expected behaviour, the evaluator examines the test plan to gain an understanding of the testing approach. In most cases, the testing approach will entail a security function being stimulated at either the external or internal interfaces and its responses are observed. However, there may be cases where a security function cannot be adequately tested at an interface (as may be the case, for instance, for residual information protection functionality); in such cases, other means will need to be employed.

7.9.1.2 Testing vs. alternate approaches to verify the expected behaviour of a security function

1112 In cases where it is impractical or inadequate to test at an interface, the test plan should identify the alternate approach to verify expected behaviour. It is the evaluator's responsibility to determine the suitability of the alternate approach. However, the following should be considered when assessing the suitability of alternate approaches:

a) an analysis of the implementation representation to determine that the required behaviour should be exhibited by the TOE is an acceptable alternate approach. This could mean a code inspection for a software TOE or perhaps a chip mask inspection for a hardware TOE.

b) it is acceptable to use evidence of developer integration or module testing, even if the EAL is not commensurate with evaluation exposure to the low-level design or implementation. If evidence of developer integration or module testing is used in verifying the expected behaviour of a security function, care should be given to confirm that the testing evidence reflects the current implementation of the TOE. If the subsystem or modules have been changed since testing occurred, evidence that the changes were tracked and addressed by analysis or further testing will usually be required.

1113 It should be emphasized that supplementing the testing effort with alternate approaches should only be undertaken when both the developer and evaluator determine that there exists no other practical means to test the expected behaviour of a security function. This alternative is made available to the developer to minimize the cost (time and/or money) of testing under the circumstances described above; it is not designed to give the evaluator more latitude to demand unwarranted additional information about the TOE, nor to replace testing in general.

7.9.1.3 Verifying the adequacy of tests

1114 Test prerequisites are necessary to establish the required initial conditions for the test. They may be expressed in terms of parameters that must be set or in terms of test ordering in cases where the completion of one test establishes the necessary prerequisites for another test. The evaluator must determine that the prerequisites are complete and appropriate in that they will not bias the observed test results towards the expected test results.

1115 The test steps and expected results specify the actions and parameters to be applied to the interfaces as well as how the expected results should be verified and what they are. The evaluator must determine that the test steps and expected results are consistent with the functional specification and the high-level design. The tests must verify behaviour documented in these specifications. This means that each security functional behaviour characteristic explicitly described in the functional specification and high-level design should have tests and expected results to verify that behaviour.

1116 Although all of the TSF has to be tested by the developer, exhaustive specification testing of the interfaces is not required. The overall aim of this activity is to determine that each security function has been sufficiently tested against the behavioural claims in the functional specification and high-level design. The test procedures will provide insight as to how the security functions have been exercised by the developer during testing. The evaluator will use this information when developing additional tests to independently test the TOE.

7.9.2 Evaluation of coverage (ATE_COV.2)

7.9.2.1 Objectives

1117 The objective of this sub-activity is to determine whether the testing (as documented) is sufficient to establish that the TSF has been systematically tested against the functional specification.

7.9.2.2 Input

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test coverage analysis.

7.9.2.3 Evaluator actions

This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_COV.2.1E.

7.9.2.3.1 Action ATE_COV.2.1E

ATE_COV.2.1C

3:ATE_COV.2-1 The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the tests identified in the test documentation and the functional specification is accurate.

1118 Correspondence may take the form of a table or matrix. In some cases mapping may be sufficient to show test correspondence. In other cases a rationale (typically prose) may have to supplement the correspondence analysis provided by the developer.

1119 Figure 7.2 displays a conceptual framework of the correspondence between security functions described in the functional specification and the tests outlined in the test documentation used to test them. Tests may involve one or multiple security functions depending on the test dependencies or the overall goal of the test being performed.

1120 The identification of the tests and the security functions presented in the test coverage analysis has to be unambiguous. The test coverage analysis will allow the evaluator to trace the identified tests back to the test documentation and the particular security function being tested back to the functional specification.

3:ATE_COV.2-2 The evaluator ***shall examine*** the test plan to determine that the testing approach for each security function of the TSF is suitable to demonstrate the expected behaviour.

1121 Guidance on this work unit can be found in:

a) Application notes, Section 7.9.1.1, Understanding the expected behaviour of the TOE;

b) Application notes, Section 7.9.1.2, Testing vs. alternate approaches to verify the expected behaviour of a security function.

3:ATE_COV.2-3 The evaluator ***shall examine*** the test procedures to determine that the test prerequisites, test steps and expected result(s) adequately test each security function.

Interp Note : The following paragraph is changed as a result of Interpretation 074.

1122 Guidance on this work unit, as it pertains to the functional specification, can be found in:

a) Application notes, Section 7.9.1.3, Verifying the adequacy of tests.

ATE_COV.2.2C

3:ATE_COV.2-4 The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

1123 All security functions and interfaces that are described in the functional specification have to be present in the test coverage analysis and mapped to tests in order for completeness to be claimed, although exhaustive specification testing of interfaces is not required. As Figure 7.2 displays, all the security functions have tests attributed to them and therefore complete test coverage is depicted in this example. Incomplete coverage would be evident if a security function was identified in the test coverage analysis and no tests could be attributed to it.

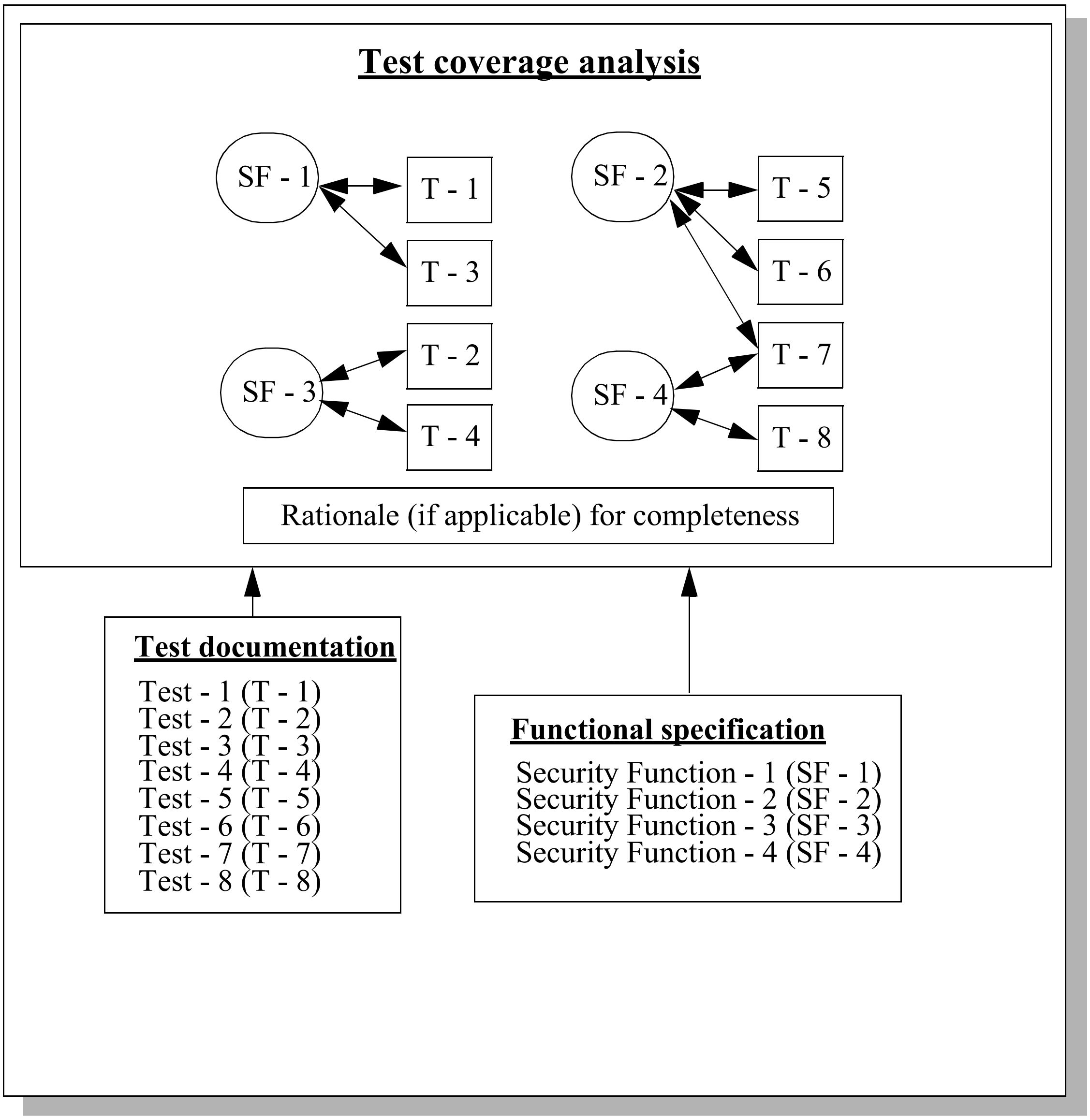

Figure 7.2 A conceptual framework of the test coverage analysis

7.9.3 Evaluation of depth (ATE_DPT.1)

7.9.3.1 Objectives

1124 The objective of this sub-activity is to determine whether the developer has tested the TSF against its high-level design.

7.9.3.2 Input

a) the ST;

b) the functional specification;

c) the high-level design;

d) the test documentation;

e) the depth of testing analysis.

7.9.3.3 Evaluator Actions

1125 This sub-activity comprises one CC part 3 evaluator action element:

a) ATE_DPT.1.1E.

7.9.3.3.1 Action ATE_DPT.1.1E

ATE_DPT.1.1C

3:ATE_DPT.1-1 The evaluator ***shall examine*** the depth of testing analysis for a mapping between the tests identified in the test documentation and the high-level design.

1126 The depth of testing analysis identifies all subsystems described in the high-level design and provides a mapping of the tests to these subsystems. Correspondence may take the form of a table or matrix. In some cases the mapping may be sufficient to show test correspondence. In other cases a rationale (typically prose) may have to supplement the mapping evidence provided by the developer.

1127 All design details specified in the high-level design that map to and satisfy TOE security requirements are subject to testing and hence, should be mapped to test documentation. Figure 7.3 displays a conceptual framework of the mapping between subsystems described in the high-level design and the tests outlined in the TOE's test documentation used to test them. Tests may involve one or multiple security functions depending on the test dependencies or the overall goal of the test being performed.

3:ATE_DPT.1-2 The evaluator ***shall examine*** the developer's test plan to determine that the testing approach for each security function of the TSF is suitable to demonstrate the expected behaviour.

1128 Guidance on this work unit can be found in:

a) Application notes, Section 7.9.1.1, Understanding the expected behaviour of the TOE;

b) Application notes, Section 7.9.1.2, Testing vs. alternate approaches to verify the expected behaviour of a security function.

1129 Testing of the TSF may be performed at the external interfaces, internal interfaces, or a combination of both. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the security functions. Specifically the evaluator determines whether testing at the internal interfaces for a security function is necessary or whether these internal interfaces can be adequately tested (albeit implicitly) by exercising the external interfaces. This determination is left to the evaluator, as is its justification.

3:ATE_DPT.1-3 The evaluator ***shall examine*** the test procedures to determine that the test pre-requisites, test steps and expected result(s) adequately test each security function.

Interp Note : The following paragraph is changed as a result of Interpretation 074.

1130 Guidance on this work unit, as it pertains to the high-level design, can be found in:

a) Application notes, Section 7.9.1.3, Verifying the adequacy of tests.

3:ATE_DPT.1-4 The evaluator ***shall check*** the depth of testing analysis to ensure that the TSF as defined in the high-level design is completely mapped to the tests in the test documentation.

1131 The depth of testing analysis provides a complete statement of correspondence between the high-level design and the test plan and procedures. All subsystems and internal interfaces described in the high-level design have to be present in the depth of testing analysis. All the subsystems and internal interfaces present in the depth of testing analysis must have tests mapped to them in order for completeness to be claimed. As Figure 7.3 displays, all the subsystems and internal interfaces have tests attributed to them and therefore complete depth of testing is depicted in this example. Incomplete coverage would be evident if a subsystem or internal interface was identified in the depth of testing analysis and no tests could be attributed to it.

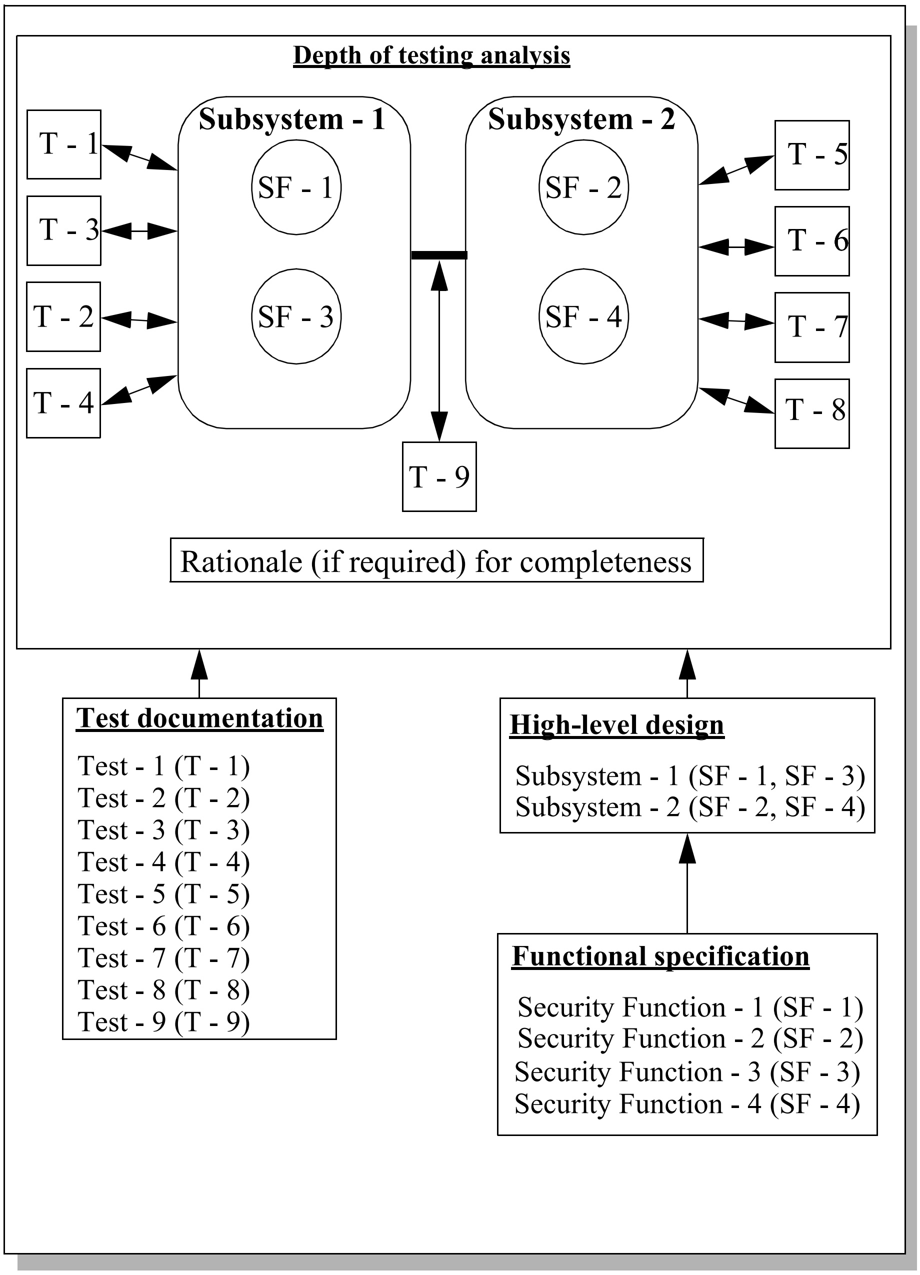

Figure 7.3 A conceptual framework of the depth of testing analysis

7.9.4 Evaluation of functional tests (ATE_FUN.1)

7.9.4.1 Objectives

1132 The objective of this sub-activity is to determine whether the developer's functional test documentation is sufficient to demonstrate that security functions perform as specified.

7.9.4.2 Application notes

1133 The extent to which the test documentation is required to cover the TSF is dependent upon the coverage assurance component.

1134 For the developer tests provided, the evaluator determines whether the tests are repeatable, and the extent to which the developer's tests can be used for the evaluator's independent testing effort. Any security function for which the developer's test results indicate that it may not perform as specified should be tested independently by the evaluator to determine whether or not it does.

7.9.4.3 Input

1135 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test procedures.

7.9.4.4 Evaluator actions

1136 This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_FUN.1.1E.

7.9.4.4.1 Action ATE_FUN.1.1E

ATE_FUN.1.1C

3:ATE_FUN.1-1 The evaluator **shall check** that the test documentation includes test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C

3:ATE_FUN.1-2 The evaluator ***shall check*** that the test plan identifies the security functions to be tested.

1137 One method that could be used to identify the security function to be tested is a reference to the appropriate part(s) of the functional specification that specifies the particular security function.

1138 The evaluator may wish to employ a sampling strategy when performing this work unit.

1139 For guidance on sampling see Annex B.2.

3:ATE_FUN.1-3 The evaluator ***shall examine*** the test plan to determine that it describes the goal of the tests performed.

1140 The test plan provides information about how the security functions are tested and the test configuration in which testing occurs.

1141 The evaluator may wish to employ a sampling strategy when performing this work unit.

1142 For guidance on sampling see Annex B.2.

3:ATE_FUN.1-4 The evaluator ***shall examine*** the test plan to determine that the TOE test configuration is consistent with the configuration identified for evaluation in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

1143 ~~The TOE used for testing should have the same unique reference as established by the ACM_CAP.3 sub-activity and the developer supplied test documentation.~~ The TOE referred to in the developer's test plan should have the same unique reference as established by the ACM_CAP.3 sub-activity.

1144 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator verifies that there are test configurations identified in the developer test documentation that are consistent with each evaluated configuration described in the ST.

1145 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

3:ATE_FUN.1-5 The evaluator ***shall examine*** the test plan to determine that it is consistent with the test procedure descriptions.

1146 The evaluator may wish to employ a sampling strategy when performing this work unit.

1147 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.3C

3:ATE_FUN.1-6 The evaluator ***shall check*** that the test procedure descriptions identify each security function behaviour to be tested.

1148 One method that may be used to identify the security function behaviour to be tested is a reference to the appropriate part(s) of the design specification that specifies the particular behaviour to be tested.

1149 The evaluator may wish to employ a sampling strategy when performing this work unit.

1150 For guidance on sampling see Annex B.2.

3:ATE_FUN.1-7 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to establish reproducible initial test conditions including ordering dependencies if any.

1151 Some steps may have to be performed to establish initial conditions. For example, user accounts need to be added before they can be deleted. An example of ordering dependencies on the results of other tests is the need to test the audit function before relying on it to produce audit records for another security mechanism such as access control. Another example of an ordering dependency would be where one test case generates a file of data to be used as input for another test case.

1152 The evaluator may wish to employ a sampling strategy when performing this work unit.

1153 For guidance on sampling see Annex B.2.

3:ATE_FUN.1-8 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to have a reproducible means to stimulate the security functions and to observe their behaviour.

1154 Stimulus is usually provided to a security function externally through the TSFI. Once an input (stimulus) is provided to the TSFI, the behaviour of the security function can then be observed at the TSFI. Reproducibility is not assured unless the test procedures contain enough detail to unambiguously describe the stimulus and the behaviour expected as a result of this stimulus.

1155 The evaluator may wish to employ a sampling strategy when performing this work unit.

1156 For guidance on sampling see Annex B.2.

3:ATE_FUN.1-9 The evaluator ***shall examine*** the test procedure descriptions to determine that they are consistent with the test procedures.

1157 If the test procedure descriptions are the test procedures, then this work unit is not applicable and is therefore considered to be satisfied.

1158 The evaluator may wish to employ a sampling strategy when performing this work unit.

1159 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.4C

3:ATE_FUN.1-10 The evaluator ***shall examine*** the test documentation to determine that sufficient expected tests results are included.

1160 The expected test results are needed to determine whether or not a test has been successfully performed. Expected test results are sufficient if they are unambiguous and consistent with expected behaviour given the testing approach.

1161 The evaluator may wish to employ a sampling strategy when performing this work unit.

1162 For guidance on sampling see Annex B.2.

ATE_FUN.1.5C

3:ATE_FUN.1-11 The evaluator ***shall check*** that the expected test results in the test documentation are consistent with the actual test results provided.

1163 A comparison of the actual and expected test results provided by the developer will reveal any inconsistencies between the results.

1164 It may be that a direct comparison of actual results cannot be made until some data reduction or synthesis has been first performed. In such cases, the developer's test documentation should describe the process to reduce or synthesize the actual data.

1165 For example, the developer may need to test the contents of a message buffer after a network connection has occurred to determine the contents of the buffer. The message buffer will contain a binary number. This binary number would have to be converted to another form of data representation in order to make the test more meaningful. The conversion of this binary representation of data into a higher-level representation will have to be described by the developer in enough detail to allow an evaluator to perform the conversion process (i.e. synchronous or asynchronous transmission, number of stop bits, parity, etc.).

1166 It should be noted that the description of the process used to reduce or synthesize the actual data is used by the evaluator not to actually perform the necessary modification but to assess whether this process is correct. It is up to the developer to transform the expected test results into a format that allows an easy comparison with the actual test results.

1167 The evaluator may wish to employ a sampling strategy when performing this work unit.

1168 For guidance on sampling see Annex B.2.

1169 If the expected and actual test results for any test are not the same, then a demonstration of the correct operation of a security function has not been achieved. Such an occurrence will influence the evaluator's independent testing effort to include testing the implicated security function. The evaluator should also consider increasing the sample of evidence upon which this work unit is performed.

3:ATE_FUN.1-12 The evaluator ***shall report*** the developer testing effort, outlining the testing approach, configuration, depth and results.

1170 The developer testing information recorded in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing of the TOE by the developer. The intent of providing this information is to give a meaningful overview of the developer testing effort. It is not intended that the information regarding developer testing in the ETR be an exact reproduction of specific test steps or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the developer's testing approach, amount of testing performed, TOE test configurations, and the overall results of the developer testing.

1171 Information that would typically be found in the ETR section regarding the developer testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested.

b) testing approach. An account of the overall developer testing strategy employed.

c) amount of developer testing performed. A description on the extent of coverage and depth of developer testing.

d) testing results. A description of the overall developer testing results.

1172 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the developer testing effort.

7.9.5 Evaluation of independent testing (ATE_IND.2)

7.9.5.1 Objectives

1173 The goal of this activity is to determine, by independently testing a subset of the TSF, whether the TOE behaves as specified, and to gain confidence in the developer's test results by performing a sample of the developer's tests.

7.9.5.2 Input

1174 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures;

f) the test documentation;

g) the test coverage analysis;

h) the depth of testing analysis;

i) the TOE suitable for testing.

7.9.5.3 Evaluator actions

1175 This sub-activity comprises three CC Part 3 evaluator action elements:

a) ATE_IND.2.1E;

b) ATE_IND.2.2E;

c) ATE_IND.2.3E.

7.9.5.3.1 Action ATE_IND.2.1E

ATE_IND.2.1C

3:ATE_IND.2-1 The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

1176 The TOE used for evaluator testing should have the same unique reference as established by the ACM_CAP.3 sub-activity ~~and the developer supplied test documentation~~.

1177 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator's TOE ~~verifies that there are~~ test configurations should be consistent with each evaluated configuration described in the ST.

1178 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

1179 If any test resources are used (e.g. meters, analysers) it will be the evaluator's responsibility to ensure that these resources are calibrated correctly.

3:ATE_IND.2-2 The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

1180 It is possible for the evaluator to determine the state of the TOE in a number of ways. For example, previous successful completion of the ADO_IGS.1 sub-activity will satisfy this work unit if the evaluator still has confidence that the TOE being used for testing was installed properly and is in a known state. If this is not the case, then the evaluator should follow the developer's procedures to install, generate and start up the TOE, using the supplied guidance only.

1181 If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed could satisfy work unit 3:ADO_IGS.1-2.

ATE_IND.2.2C

3:ATE_IND.2-3 The evaluator ***shall examine*** the set of resources provided by the developer to determine that they are equivalent to the set of resources used by the developer to functionally test the TSF.

1182 The resource set may include laboratory access and special test equipment, among others. Resources that are not identical to those used by the developer need to be equivalent in terms of any impact they may have on test results.

7.9.5.3.2 Action ATE_IND.2.2E

3:ATE_IND.2-4 The evaluator ***shall devise*** a test subset.

1183 The evaluator selects a test subset and testing strategy that is appropriate for the TOE. One extreme testing strategy would be to have the test subset contain as many security functions as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few security functions based on their perceived relevance and rigorously test these functions.

1184 Typically the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the security functional requirements identified in the ST using at least one test, but testing need not demonstrate exhaustive specification testing.

1185 The evaluator, when selecting the subset of the TSF to be tested, should consider the following factors:

a) The developer test evidence. The developer test evidence consists of: the test coverage analysis, the depth of testing analysis, and the test documentation. The developer test evidence will provide insight as to how the security functions have been exercised by the developer during testing. The evaluator applies this information when developing new tests to independently test the TOE. Specifically the evaluator should consider:

    1) augmentation of developer testing for specific security function(s). The evaluator may wish to perform more of the same type of tests by varying parameters to more rigorously test the security function.

    2) supplementation of developer testing strategy for specific security function(s). The evaluator may wish to vary the testing approach of a specific security function by testing it using another test strategy.

b) The number of security functions from which to draw upon for the test subset. Where the TOE includes only a small number of security functions, it may be practical to rigourously test all of the security functions. For TOEs with a large number of security functions this will not be cost-effective, and sampling is required.

c) Maintaining a balance of evaluation activities. The evaluator effort expended on the test activity should be commensurate with that expended on any other evaluation activity.

1186 The evaluator selects the security functions to compose the subset. This selection will depend on a number of factors, and consideration of these factors may also influence the choice of test subset size:

a) Rigour of developer testing of the security functions. All security functions identified in the functional specification had to have developer test evidence attributed to them as required by ATE_COV.2. Those security functions that the evaluator determines require additional testing should be included in the test subset.

b) Developer test results. If the results of developer tests cause the evaluator to doubt that a security function, or aspect thereof, operates as specified, then the evaluator should include such security functions in the test subset.

c) Known public domain weaknesses commonly associated with the type of TOE (e.g. operating system, firewall). Know public domain weaknesses associated with the type of TOE will influence the selection process of the test subset. The evaluator should include those security functions that address known public domain weaknesses for that type of TOE in the subset (know public domain weaknesses in this context does not refer to vulnerabilities as such but to inadequacies or problem areas that have been experienced with this particular type of TOE). If no such weaknesses are known, then a more general approach of selecting a broad range of security functions may be more appropriate.

d) Significance of security functions. Those security functions more significant than others in terms of the security objectives for the TOE should be included in the test subset.

e) SOF claims made in the ST. All security functions for which a specific SOF claim has been made should be included in the test subset.

f) Complexity of the security function. Complex security functions may require complex tests that impose onerous requirements on the developer or evaluator, which will not be conducive to cost-effective evaluations. Conversely, complex security functions are a likely area to find errors and are good candidates for the subset. The evaluator will need to strike a balance between these considerations.

g) Implicit testing. Testing some security functions may often implicitly test other security functions, and their inclusion in the subset may maximize the number of security functions tested (albeit implicitly). Certain interfaces will typically be used to provide a variety of security functionality, and will tend to be the target of an effective testing approach.

h) Types of interfaces to the TOE (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

i) Functions that are innovative or unusual. Where the TOE contains innovative or unusual security functions, which may feature strongly in marketing literature, these should be strong candidates for testing.

1187 This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

1188 For guidance on sampling see Annex B.2.

3:ATE_IND.2-5 The evaluator ***shall produce*** test documentation for the test subset that is sufficiently detailed to enable the tests to be reproducible.

1189 With an understanding of the expected behaviour of a security function, from the ST and the functional specification, the evaluator has to determine the most feasible way to test the function. Specifically the evaluator considers:

a) the approach that will be used, for instance, whether the security function will be tested at an external interface, at an internal interface using a test harness, or will an alternate test approach be employed (e.g. in exceptional circumstances, a code inspection);

b) the security function interface(s) that will be used to stimulate the security function and observe responses;

c) the initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

d) special test equipment that will be required to either stimulate a security function (e.g. packet generators) or make observations of a security function (e.g. network analysers).

1190 The evaluator may find it practical to test each security function using a series of test cases, where each test case will test a very specific aspect of expected behaviour.

1191 The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant design specification, and to the ST, if necessary.

3:ATE_IND.2-6 The evaluator ***shall conduct*** testing.

1192 The evaluator uses the test documentation developed as a basis for executing tests on the TOE. The test documentation is used as a basis for testing but this does not preclude the evaluator from performing additional ad hoc tests. The evaluator may devise new tests based on behaviour of the TOE discovered during testing. These new tests are recorded in the test documentation.

3:ATE_IND.2-7 The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the security function behaviour to be tested;

b) instructions to connect and setup all required test equipment as required to conduct the test;

c) instructions to establish all prerequisite test conditions;

d) instructions to stimulate the security function;

e) instructions for observing the behaviour of the security function;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE;

h) actual test results.

1193 The level of detail should be such that another evaluator could repeat the tests and obtain an equivalent result. While some specific details of the test results may be different (e.g. time and date fields in an audit record) the overall result should be identical.

1194 There may be instances when it is unnecessary to provide all the information presented in this work unit (e.g. the actual test results of a test may not require any analysis before a comparison between the expected results can be made). The determination to omit this information is left to the evaluator, as is the justification.

3:ATE_IND.2-8 The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

1195 Any differences in the actual and expected test results may indicate that the TOE does not perform as specified or that the evaluator test documentation may be incorrect. Unexpected actual results may require corrective maintenance to the TOE or test documentation and perhaps require re-running of impacted tests and modifying the test sample size and composition. This determination is left to the evaluator, as is its justification.

7.9.5.3.3 Action ATE_IND.2.3E

3:ATE_IND.2-9 The evaluator ***shall conduct*** testing using a sample of tests found in the developer test plan and procedures.

1196 The overall aim of this work unit is to perform a sufficient number of the developer tests to confirm the validity of the developer's test results. The evaluator has to decide on the size of the sample, and the developer tests that will compose the sample.

1197 Taking into consideration the overall evaluator effort for the entire tests activity, normally 20% of the developer's tests should be performed although this may vary according to the nature of the TOE, and the test evidence supplied.

1198 All the developer tests can be traced back to specific security function(s). Therefore, the factors to consider in the selection of the tests to compose the sample are similar to those listed for subset selection in work-unit ATE_IND.2-4. Additionally, the evaluator may wish to employ a random sampling method to select developer tests to include in the sample.

1199 For guidance on sampling see Annex B.2.

3:ATE_IND.2-10 The evaluator ***shall check*** that all the actual test results are consistent with the expected test results.

1200 Inconsistencies between the developer's expected test results and actual test results will compel the evaluator to resolve the discrepancies. Inconsistencies encountered by the evaluator could be resolved by a valid explanation and resolution of the inconsistencies by the developer.

1201 If a satisfactory explanation or resolution can not be reached, the evaluator's confidence in the developer's test results may be lessened and it may even be necessary for the evaluator to increase the sample size, to regain confidence in the developer testing. If the increase in sample size does not satisfy the evaluator's concerns, it may be necessary to repeat the entire set of developer's tests.Ultimately, to the extent that the TSF subset identified in work unit ATE_IND.2-4 is adequately tested, deficiencies with the developer's tests need to result in either corrective action to the developer's tests or in the production of new tests by the evaluator.

3:ATE_IND.2-11 The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

1202 The evaluator testing information reported in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing activity during the evaluation. The intent of providing this information is to give a meaningful overview of the testing effort. It is not intended that the information regarding testing in the ETR be an exact reproduction of specific test instructions or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the testing approach chosen, amount of evaluator testing performed, amount of developer tests performed, TOE test configurations, and the overall results of the testing activity.

1203 Information that would typically be found in the ETR section regarding the evaluator testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested.

b) subset size chosen. The amount of security functions that were tested during the evaluation and a justification for the size.

c) selection criteria for the security functions that compose the subset. Brief statements about the factors considered when selecting security functions for inclusion in the subset.

d) security functions tested. A brief listing of the security functions that merited inclusion in the subset.

e) developer tests performed. The amount of developer tests performed and a brief description of the criteria used to select the tests.

f) verdict for the activity. The overall judgement on the results of testing during the evaluation.

This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the testing the evaluator performed during the evaluation.

7.10 Vulnerability assessment activity

1204 The purpose of the vulnerability assessment activity is to determine the existence and exploitability of flaws or weaknesses in the TOE in the intended environment. This determination is based upon analysis performed by the developer and the evaluator, and is supported by evaluator testing.

1205 The vulnerability assessment activity at EAL3 contains sub-activities related to the following components:

a) AVA_MSU.1;

b) AVA_SOF.1;

c) AVA_VLA.1.

7.10.1 Evaluation of misuse (AVA_MSU.1)

7.10.1.1 Objectives

1206 The objectives of this sub-activity are to determine whether the guidance is misleading, unreasonable or conflicting, whether secure procedures for all modes of operation have been addressed, and whether use of the guidance will facilitate prevention and detection of insecure TOE states.

7.10.1.2 Application notes

1207 The use of the term *guidance* in this sub-activity refers to the user guidance, the administrator guidance, and the secure installation, generation, and start-up procedures. Installation, generation, and start-up procedures here refers to all procedures the administrator is responsible to perform to progress the TOE from a delivered state to an operational state.

7.10.1.3 Input

1208 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures;

g) the test documentation.

7.10.1.4 Evaluator actions

1209 This sub-activity comprises three CC Part 3 evaluator action elements:

a) AVA_MSU.1.1E;

b) AVA_MSU.1.2E;

c) AVA_MSU.1.3E.

7.10.1.4.1 Action AVA_MSU.1.1E

AVA_MSU.1.1C

3:AVA_MSU.1-1 The evaluator ***shall examine*** the guidance and other evaluation evidence to determine that the guidance identifies all possible modes of operation of the TOE (including, if applicable, operation following failure or operational error), their consequences and implications for maintaining secure operation.

1210 Other evaluation evidence, particularly the functional specification and test documentation, provide an information source that the evaluator should use to determine that the guidance contains sufficient guidance information.

1211 The evaluator should focus on a single security function at a time, comparing the guidance for securely using the security function with other evaluation evidence, to determine that the guidance related to the security function is sufficient for the secure usage (i.e. consistent with the TSP) of that security function. The evaluator should also consider the relationships between functions, searching for potential conflicts.

AVA_MSU.1.2C

3:AVA_MSU.1-2 The evaluator ***shall examine*** the guidance to determine that it is clear and internally consistent.

1212 The guidance is unclear if it can reasonably be misconstrued by an administrator or user, and used in a way detrimental to the TOE, or to the security provided by the TOE.

1213 For guidance on consistency analysis see Annex B.3.

3:AVA_MSU.1-3 The evaluator ***shall examine*** the guidance and other evaluation evidence to determine that the guidance is complete and reasonable.

1214 The evaluator should apply familiarity with the TOE gained from performing other evaluation activities to determine that the guidance is complete.

1215 In particular, the evaluator should consider the functional specification and TOE summary specification. All security functions described in these documents should be described in the guidance as required to permit their secure administration and use. The evaluator may, as an aid, prepare an informal mapping between the guidance and these documents. Any omissions in this mapping may indicate incompleteness.

1216 The guidance is unreasonable if it makes demands on the TOE's usage or operational environment that are inconsistent with the ST or unduly onerous to maintain security.

1217 The evaluator should note that results gained during the performance of work units from the AGD_ADM sub-activity will provide useful input to this examination.

AVA_MSU.1.3C

3:AVA_MSU.1-4 The evaluator ***shall examine*** the guidance to determine that all assumptions about the intended environment are articulated.

1218 The evaluator analyses the assumptions about the intended TOE security environment of the ST and compares them with the guidance to ensure that all assumptions about the intended TOE security environment of the ST that are relevant to the administrator or user are described appropriately in the guidance.

AVA_MSU.1.4C

3:AVA_MSU.1-5 The evaluator ***shall examine*** the guidance to determine that all requirements for external security measures are articulated.

1219 The evaluator analyses the guidance to ensure that it lists all external procedural, physical, personnel and connectivity controls. The security objectives in the ST for the non-IT environment will indicate what is required.

7.10.1.4.2 Action AVA_MSU.1.2E

3:AVA_MSU.1-6 The evaluator ***shall perform*** all administrator and user (if applicable) procedures necessary to configure and install the TOE to determine that the TOE can be configured and used securely using only the supplied guidance.

1220 Configuration and installation requires the evaluator to advance the TOE from a deliverable state to the state in which the TOE is operational and enforcing a TSP consistent with the security objectives specified in the ST.

1221 The evaluator should follow only the developer's procedures as documented in the user and administrator guidance that is normally supplied to the consumer of the TOE. Any difficulties encountered during such an exercise may be indicative of incomplete, unclear, inconsistent or unreasonable guidance.

1222 Note that work performed to satisfy this work unit may also contribute towards satisfying evaluator action ADO_IGS.1.2E.

7.10.1.4.3 Action AVA_MSU.1.3E

3:AVA_MSU.1-7 The evaluator ***shall examine*** the guidance to determine that sufficient guidance is provided for the consumer to effectively administer and use the TOE's security functions, and to detect insecure states.

1223 TOEs may use a variety of ways to assist the consumer in effectively using the TOE securely. One TOE may employ functionality (features) to alert the consumer when the TOE is in an insecure state, whilst other TOEs may be delivered with enhanced guidance containing suggestions, hints, procedures, etc. on using the existing security features most effectively; for instance, guidance on using the audit feature as an aid for detecting insecure states.

1224 To arrive at a verdict for this work unit, the evaluator considers the TOE's functionality, its purpose and intended environment, and assumptions about its usage or users. The evaluator should arrive at the conclusion that, if the TOE can transition into an insecure state, there is reasonable expectation that use of the guidance would permit the insecure state to be detected in a timely manner. The potential for the TOE to enter into insecure states may be determined using the evaluation deliverables, such as the ST, the functional specification and the high-level design of the TSF.

7.10.2 Evaluation of strength of TOE security functions (AVA_SOF.1)

7.10.2.1 Objectives

1225 The objectives of this sub-activity are to determine whether SOF claims are made in the ST for all probabilistic or permutational mechanisms and whether the developer's SOF claims made in the ST are supported by an analysis that is correct.

7.10.2.2 Application notes

1226 SOF analysis is performed on mechanisms that are probabilistic or permutational in nature, such as password mechanisms or biometrics. Although cryptographic mechanisms are also probabilistic in nature and are often described in terms of *strength*, AVA_SOF.1 is not applicable to cryptographic mechanisms. For such mechanisms, the evaluator should seek scheme guidance.

1227 Although SOF analysis is performed on the basis of individual mechanisms, the overall determination of SOF is based on functions. Where more than one probabilistic or permutational mechanism is employed to provide a security function, each distinct mechanism must be analysed. The manner in which these mechanisms combine to provide a security function will determine the overall SOF level for that function. The evaluator needs design information to understand how the mechanisms work together to provide a function, and a minimum level for such information is given by the dependency on ADV_HLD.1. The actual design information available to the evaluator is determined by the EAL, and the available information should be used to support the evaluator's analysis when required.

1228 For a discussion on SOF in relation to multiple TOE domains see Section 4.4.6.

7.10.2.3 Input

1229 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the strength of TOE security functions analysis.

7.10.2.4 Evaluator actions

1230 This sub-activity comprises two CC Part 3 evaluator action elements:

a) AVA_SOF.1.1E;

b) AVA_SOF.1.2E.

7.10.2.4.1 Action AVA_SOF.1.1E

AVA_SOF.1.1C

3:AVA_SOF.1-1 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a SOF rating.

1231 If SOF claims are expressed solely as SOF metrics, then this work unit is not applicable and is therefore considered to be satisfied.

1232 A SOF rating is expressed as one of SOF-basic, SOF-medium or SOF-high, which are defined in terms of attack potential - refer to the CC Part 1 Glossary. A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational security mechanisms. However, individual mechanisms may have a SOF claim expressed as a rating that exceeds the overall SOF requirement.

1233 Guidance on determining the attack potential necessary to effect an attack and, hence, to determine SOF as a rating is in Annex B.8.

1234 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.2C

3:AVA_SOF.1-2 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a metric.

1235 If SOF claims are expressed solely as SOF ratings, then this work unit is not applicable and is therefore considered to be satisfied.

1236 A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational mechanisms. However, individual mechanisms may have a SOF claim expressed as a metric that meets or exceeds the overall SOF requirement.

1237 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.1C and AVA_SOF.1.2C

3:AVA_SOF.1-3 The evaluator ***shall examine*** the SOF analysis to determine that any assertions or assumptions supporting the analysis are valid.

1238 For example, it may be a flawed assumption that a particular implementation of a pseudo-random number generator will possess the required entropy necessary to seed the security mechanism to which the SOF analysis is relevant.

1239 Assumptions supporting the SOF analysis should reflect the *worst case*, unless *worst case* is invalidated by the ST. Where a number of different possible scenarios exist, and these are dependent on the behaviour of the human user or attacker, the case that represents the lowest strength should be assumed unless, as previously stated, this case is invalid.

1240 For example, a strength claim based upon a maximum theoretical password space (i.e. all printable ASCII characters) would not be *worst case* because it is human behaviour to use natural language passwords, effectively reducing the password space and associated strength. However, such an assumption could be appropriate if the TOE used IT measures, identified in the ST, such as password filters to minimise the use of natural language passwords.

3:AVA_SOF.1-4 The evaluator ***shall examine*** the SOF analysis to determine that any algorithms, principles, properties and calculations supporting the analysis are correct.

1241 The nature of this work unit is highly dependent upon the type of mechanism being considered. Annex B.8 provides an example SOF analysis for an identification and authentication function that is implemented using a password mechanism; the analysis considers the maximum password space to ultimately arrive at a SOF rating. For biometrics, the analysis should consider resolution and other factors impacting the mechanism's susceptibility to spoofing.

1242 SOF expressed as a rating is based on the minimum attack potential required to defeat the security mechanism. The SOF ratings are defined in terms of attack potential in CC Part 1 Glossary.

1243 For guidance on attack potential see Annex B.8.

3:AVA_SOF.1-5 The evaluator ***shall examine*** the SOF analysis to determine that each SOF claim is met or exceeded.

1244 For guidance on the rating of SOF claims see Annex B.8.

3:AVA_SOF.1-6 The evaluator ***shall examine*** the SOF analysis to determine that all functions with a SOF claim meet the minimum strength level defined in the ST.

7.10.2.4.2 Action AVA_SOF.1.2E

3:AVA_SOF.1-7 The evaluator ***shall examine*** the functional specification, the high-level design, the user guidance and the administrator guidance to determine that all probabilistic or permutational mechanisms have a SOF claim.

1245 The identification by the developer of security functions that are realised by probabilistic or permutational mechanisms is verified during the ST evaluation. However, because the TOE summary specification may have been the only evidence available upon which to perform that activity, the identification of such mechanisms may be incomplete. Additional evaluation evidence required as input to this sub-activity may identify additional probabilistic or permutational mechanisms not already identified in the ST. If so, the ST will have to be updated appropriately to reflect the additional SOF claims and the developer will need to provide additional analysis that justifies the claims as input to evaluator action AVA_SOF.1.1E.

3:AVA_SOF.1-8 The evaluator ***shall examine*** the SOF claims to determine that they are correct.

1246 Where the SOF analysis includes assertions or assumptions (e.g. about how many authentication attempts are possible per minute), the evaluator should independently confirm that these are correct. This may be achieved through testing or through independent analysis.

7.10.3 Evaluation of vulnerability analysis (AVA_VLA.1)

7.10.3.1 Objectives

1247 The objective of this sub-activity is to determine whether the TOE, in its intended environment, has exploitable obvious vulnerabilities.

7.10.3.2 Application notes

1248 The use of the term *guidance* in this sub-activity refers to the user guidance, the administrator guidance, and the secure installation, generation, and start-up procedures.

1249 The consideration of exploitable vulnerabilities will be determined by the security objectives and functional requirements in the ST. For example, if measures to prevent bypass of the security functions are not required in the ST (FPT_PHP, FPT_RVM and FPT_SEP are absent) then vulnerabilities based on bypass should not be considered.

1250 Vulnerabilities may be in the public domain, or not, and may require skill to exploit, or not. These two aspects are related, but are distinct. It should not be assumed that, simply because a vulnerability is in the public domain, it can be easily exploited.

1251 The following terms are used in the guidance with specific meaning:

a) Vulnerability - a weakness in the TOE that can be used to violate a security policy in some environment;

b) Vulnerability analysis - A systematic search for vulnerabilities in the TOE, and an assessment of those found to determine their relevance for the intended environment for the TOE;

c) Obvious vulnerability - a vulnerability that is open to exploitation that requires a minimum of understanding of the TOE, technical sophistication and resources;

d) Potential vulnerability - A vulnerability the existence of which is suspected (by virtue of a postulated attack path), but not confirmed, in the TOE;

e) Exploitable vulnerability - A vulnerability that can be exploited in the intended environment for the TOE;

f) Non-exploitable vulnerability - A vulnerability that cannot be exploited in the intended environment for the TOE;

g) Residual vulnerability - A non-exploitable vulnerability that could be exploited by an attacker with greater attack potential than is anticipated in the intended environment for the TOE;

h) Penetration testing - Testing carried out to determine the exploitability of identified TOE potential vulnerabilities in the intended environment for the TOE.

7.10.3.3 Input

1252 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures;

g) the vulnerability analysis;

h) the strength of function claims analysis;

i) the TOE suitable for testing.

1253 Other input for this sub-activity is:

a) current information regarding obvious vulnerabilities (e.g. from an overseer).

7.10.3.4 Evaluator actions

1254 This sub-activity comprises two CC Part 3 evaluator action elements:

a) AVA_VLA.1.1E;

b) AVA_VLA.1.2E.

7.10.3.4.1 Action AVA_VLA.1.1E

Interp Note : The following citation is changed as a result of Interpretation 051.

AVA_VLA.1.1C, AVA_VLA.1.2C and AVA_VLA.1.3C

3:AVA_VLA.1-1 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that the search for obvious vulnerabilities has considered all relevant information.

1255 The developer's vulnerability analysis should cover the developer's search for obvious vulnerabilities in at least all evaluation deliverables and public domain information sources. The evaluator should use the evaluation deliverables, not to perform an independent vulnerability analysis (not required at AVA_VLA.1), but as a basis for assessing the developer's search for obvious vulnerabilities.

Interp Note : The following paragraph is added as a result of Interpretation 031.

Information in the public domain is highly dynamic. Therefore, it is possible that new vulnerabilities are reported in the public domain between the time the developer performs the vulnerability analysis and the time that the evaluation is completed. The point at which monitoring of the public domain information ceases is an evaluation authority issue; therefore guidance and agreement should be sought from the evaluation authority.

3:AVA_VLA.1-2 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that each obvious vulnerability is described and that a rationale is given for why it is not exploitable in the intended environment for the TOE.

1256 The developer is expected to search for obvious vulnerabilities, based on knowledge of the TOE, and of public domain information sources. Given the requirement to identify only obvious vulnerabilities, a detailed analysis is not expected. The developer filters this information, based on the above definition, and shows that obvious vulnerabilities are not exploitable in the intended environment.

1257 The evaluator needs to be concerned with three aspects of the developer's analysis:

a) whether the developer's analysis has considered all evaluation deliverables;

b) whether appropriate measures are in place to prevent the exploitation of obvious vulnerabilities in the intended environment;

c) whether some obvious vulnerabilities remain unidentified.

1258 The evaluator should not be concerned over whether identified vulnerabilities are obvious or not, unless this is used by the developer as a basis for determining non-exploitability. In such a case the evaluator validates the assertion by determining resistance to an attacker with low attack potential for the identified vulnerability.

1259 The concept of *obvious vulnerabilities* is not related to that of *attack potential*. The latter is determined by the evaluator during independent vulnerability analysis. Since this activity is not performed for AVA_VLA.1, there is normally no searching and filtering by the evaluator on the basis of attack potential. However, the evaluator may still discover potential vulnerabilities during the evaluation, and the determination of how these should be addressed will be made by reference to the definition of obvious vulnerabilities and the concept of low attack potential.

1260 The determination as to whether some obvious vulnerabilities remain unidentified is limited to assessment of the validity of the developer's analysis, a comparison with available public domain vulnerability information, and a comparison with any further vulnerabilities identified by the evaluator during the course of other evaluation activities.

1261 A vulnerability is termed non-exploitable if one or more of the following conditions exist:

a) security functions or measures in the (IT or non-IT) environment prevent exploitation of the vulnerability in the intended environment. For instance, restricting physical access to the TOE to authorised users only may effectively render a TOE's vulnerability to tampering unexploitable;

b) the vulnerability is exploitable but only by attackers possessing moderate or high attack potential. For instance, a vulnerability of a distributed TOE to session hijack attacks requires an attack potential beyond that required to exploit an obvious vulnerability. However, such vulnerabilities are reported in the ETR as residual vulnerabilities.

c) either the threat is not claimed to be countered or the violable organisational security policy is not claimed to be achieved by the ST. For instance, a firewall whose ST makes no availability policy claim and is vulnerable to TCP SYN attacks (an attack on a common Internet protocol that renders hosts incapable of servicing connection requests) should not fail this evaluator action on the basis of this vulnerability alone.

1262 For guidance on determining attack potential necessary to exploit a vulnerability see Annex B.8.

3:AVA_VLA.1-3 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that it is consistent with the ST and the guidance.

1263 The developer's vulnerability analysis may address a vulnerability by suggesting specific configurations or settings for TOE functions. If such operating constraints are deemed to be effective and consistent with the ST, then all such configurations/ settings should be adequately described in the guidance so that they may be employed by the consumer.

7.10.3.4.2 Action AVA_VLA.1.2E

3:AVA_VLA.1-4 The evaluator ***shall devise*** penetration tests, building on the developer vulnerability analysis.

1264 The evaluator prepares for penetration testing:

a) as necessary to attempt to disprove the developer's analysis in cases where the developer's rationale for why a vulnerability is unexploitable is suspect in the opinion of the evaluator;

b) as necessary to determine the susceptibility of the TOE, in its intended environment, to an obvious vulnerability not considered by the developer. The evaluator should have access to current information (e.g. from the overseer) regarding obvious public domain vulnerabilities that may not have been considered by the developer, and may also have identified potential vulnerabilities as a result of performing other evaluation activities.

1265 The evaluator is not expected to test for vulnerabilities (including those in the public domain) beyond those which are obvious. In some cases, however, it will be necessary to carry out a test before the exploitability can be determined. Where, as a result of evaluation expertise, the evaluator discovers a vulnerability that is beyond obvious, this is reported in the ETR as a residual vulnerability.

1266 With an understanding of the suspected obvious vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically the evaluator considers:

a) the security function interfaces that will be used to stimulate the TSF and observe responses;

b) initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

c) special test equipment that will be required to either stimulate a security function or make observations of a security function (although it is unlikely that specialist equipment would be required to exploit an obvious vulnerability).

1267 The evaluator will probably find it practical to carry out penetration testing using a series of test cases, where each test case will test for a specific obvious vulnerability.

3:AVA_VLA.1-5 The evaluator ***shall produce*** penetration test documentation for the tests that build upon the developer vulnerability analysis, in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the obvious vulnerability the TOE is being tested for;

b) instructions to connect and setup all required test equipment as required to conduct the penetration test;

c) instructions to establish all penetration test prerequisite initial conditions;

d) instructions to stimulate the TSF;

e) instructions for observing the behaviour of the TSF;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE.

1268 The intent of specifying this level of detail in the test documentation is to allow another evaluator to repeat the tests and obtain an equivalent result.

3:AVA_VLA.1-6 The evaluator ***shall conduct*** penetration testing, building on the developer vulnerability analysis.

1269 The evaluator uses the penetration test documentation resulting from work unit 3:AVA_VLA.1-4 as a basis for executing penetration tests on the TOE, but this does not preclude the evaluator from performing additional ad hoc penetration tests. If required, the evaluator may devise ad hoc tests as a result of information learned during penetration testing that, if performed by the evaluator, are to be recorded in the penetration test documentation. Such tests may be required to follow up unexpected results or observations, or to investigate potential vulnerabilities suggested to the evaluator during the pre-planned testing.

3:AVA_VLA.1-7 The evaluator ***shall record*** the actual results of the penetration tests.

1270 While some specific details of the actual test results may be different from those expected (e.g. time and date fields in an audit record) the overall result should be identical. Any differences should be justified.

3:AVA_VLA.1-8 The evaluator ***shall examine*** the results of all penetration testing and the conclusions of all vulnerability analysis to determine that the TOE, in its intended environment, has no exploitable obvious vulnerabilities.

1271 If the results reveal that the TOE has obvious vulnerabilities, exploitable in its intended environment, then this results in a failed verdict for the evaluator action.

3:AVA_VLA.1-9 The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

1272 The penetration testing information reported in the ETR allows the evaluator to convey the overall penetration testing approach and effort expended on this sub-activity. The intent of providing this information is to give a meaningful overview of the evaluator's penetration testing effort. It is not intended that the information regarding penetration testing in the ETR be an exact reproduction of specific test steps or results of individual penetration tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the penetration testing approach chosen, amount of penetration testing performed, TOE test configurations, and the overall results of the penetration testing activity.

1273 Information that would typically be found in the ETR section regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) security functions penetration tested. A brief listing of the security functions that were the focus of the penetration testing;

c) verdict for the sub-activity. The overall judgement on the results of penetration testing.

1274 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the penetration testing the evaluator performed during the evaluation.

3:AVA_VLA.1-10 The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. CEM activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the implicated security function(s), objective(s) not met, organisational security policy(ies) contravened and threat(s) realised;

c) a description;

d) whether it is exploitable in its intended environment or not (i.e. exploitable or residual);

e) identification of evaluation party (e.g. developer, evaluator) who identified it.

Chapter 8

EAL4 evaluation

8.1 Introduction

1275 EAL4 provides a moderate to high level of assurance. The security functions are analysed using a functional specification, guidance documentation, the high-level and low-level design of the TOE, and a subset of the implementation to understand the security behaviour. The analysis is supported by independent testing of a subset of the TOE security functions, evidence of developer testing based on the functional specification and the high level design, selective confirmation of the developer test results, analysis of strengths of the functions, evidence of a developer search for vulnerabilities, and an independent vulnerability analysis demonstrating resistance to low attack potential penetration attackers. Further assurance is gained through the use of an informal model of the TOE security policy and through the use of development environment controls, automated TOE configuration management, and evidence of secure delivery procedures.

8.2 Objectives

1276 The objective of this chapter is to define the minimal evaluation effort for achieving an EAL4 evaluation and to provide guidance on ways and means of accomplishing the evaluation.

8.3 EAL4 evaluation relationships

1277 An EAL4 evaluation covers the following:

a) evaluation input task (Chapter 2);

b) EAL4 evaluation activities comprising the following:

   1) evaluation of the ST (Chapter 4);

   2) evaluation of the configuration management (Section 8.4);

   3) evaluation of the delivery and operation documents (Section 8.5);

   4) evaluation of the development documents (Section 8.6);

   5) evaluation of the guidance documents (Section 8.7);

   6) evaluation of the life cycle support (Section 8.8);

7) evaluation of the tests (Section 8.9);

8) testing (Section 8.9);

9) evaluation of the vulnerability assessment (Section 8.10);

c) evaluation output task (Chapter 2).

1278 The evaluation activities are derived from the EAL4 assurance requirements contained in the CC Part 3.

1279 The ST evaluation is started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities.

1280 The sub-activities comprising an EAL4 evaluation are described in this chapter. Although the sub-activities can, in general, be started more or less coincidentally, some dependencies between sub-activities have to be considered by the evaluator.

1281 For guidance on dependencies see Annex B.5.

8.4 Configuration management activity

1282 The purpose of the configuration management activity is to assist the consumer in identifying the evaluated TOE, to ensure that configuration items are uniquely identified, and the adequacy of the procedures that are used by the developer to control and track changes that are made to the TOE. This includes details on what changes are tracked, how potential changes are incorporated, and the degree to which automation is used to reduce the scope for error.

1283 The configuration management activity at EAL4 contains sub-activities related to the following components:

a) ACM_AUT.1;

b) ACM_CAP.4;

c) ACM_SCP.2.

8.4.1 Evaluation of CM automation (ACM_AUT.1)

8.4.1.1 Objective

1284 The objective of this sub-activity is to determine whether changes to the implementation representation are controlled with the support of automated tools, thus making the CM system less susceptible to human error or negligence.

8.4.1.2 Input

1285 The evaluation evidence for this sub-activity is:

a) the configuration management documentation.

8.4.1.3 Evaluator actions

1286 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ACM_AUT.1.1E;

b) implied evaluator action based on ACM_AUT.1.1D.

8.4.1.3.1 Action ACM_AUT.1.1E

ACM_AUT.1.1C

4:ACM_AUT.1-1 The evaluator ***shall check*** the CM plan for a description of the automated measures to control access to the TOE implementation representation.

4:ACM_AUT.1-2 The evaluator ***shall examine*** the automated access control measures to determine that they are effective in preventing unauthorised modification of the TOE implementation representation.

1287 The evaluator reviews the configuration management documentation to identify those individuals or roles authorised to make changes to the TOE implementation representation. For example, once it is under configuration management, access to an element of the implementation representation may only be allowed for the individual who performs the software integration role.

1288 The evaluator should exercise the automated access control measures to determine whether they can be bypassed by an unauthorised role or user. This determination need only comprise a few basic tests.

ACM_AUT.1.2C

4:ACM_AUT.1-3 The evaluator ***shall check*** the CM documentation for automated means to support generation of the TOE from its implementation representation.

1289 In this work unit the term *generation* applies to those processes adopted by the developer to progress the TOE from its implementation to a state ready to be delivered to the end customer.

1290 The evaluator should verify the existence of automated generation support procedures within the CM documentation.

4:ACM_AUT.1-4 The evaluator ***shall examine*** the automated generation procedures to determine that they can be used to support generation of the TOE.

1291 The evaluator determines that by following the generation procedures a TOE would be generated that reflects its implementation representation. The customer can then be confident that the version of the TOE delivered for installation implements the TSP as described in the ST. For example, in a software TOE this may include checking that the automated generation procedures help to ensure that all source files and related libraries that are relied upon to enforce the TSP are included in the compiled object code.

1292 It should be noted that this requirement is only to provide support. For example, an approach that placed Unix makefiles under configuration management should be sufficient to meet the aim, given that in such a case automation would have made a significant contribution to accurate generation of the TOE. Automated procedures can assist in identifying the correct configuration items to be used in generating the TOE.

ACM_AUT.1.3C

4:ACM_AUT.1-5 The evaluator ***shall check*** that the CM plan includes information on the automated tools used in the CM system.

ACM_AUT.1.4C

4:ACM_AUT.1-6 The evaluator ***shall examine*** the information relating to the automated tools provided in the CM plan to determine that it describes how they are used.

1293 The information provided in the CM plan provides the necessary detail for a user of the CM system to be able to operate the automated tools correctly in order to maintain the integrity of the TOE. For example, the information provided may include a description of:

a) the functionality provided by the tools;

b) how this functionality is used by the developer to control changes to the implementation representation;

c) how this functionality is used by the developer to support generation of the TOE.

8.4.1.3.2 Implied evaluator action

ACM_AUT.1.1D

4:ACM_AUT.1-7 The evaluator ***shall examine*** the CM system to determine that the automated tools and procedures described in the CM plan are used.

1294 This work unit may be viewed as an additional activity to be carried out in parallel with the evaluator's examination into the use of the CM system required by ACM_CAP. The evaluator looks for evidence that the tools and procedures are in use. This should include a visit to the development site to witness operation of the tools and procedures, and an examination of evidence produced through their use.

1295 For guidance on site visits see Annex B.5.

8.4.2 Evaluation of CM capabilities (ACM_CAP.4)

8.4.2.1 Objectives

1296 The objectives of this sub-activity are to determine whether the developer has clearly identified the TOE and its associated configuration items, and whether the ability to modify these items is properly controlled.

8.4.2.2 Input

1297 The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

8.4.2.3 Evaluator actions

1298 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_CAP.4.1E;

8.4.2.3.1 Action ACM_CAP.4.1E

ACM_CAP.4.1C

4:ACM_CAP.4-1 The evaluator ***shall check*** that the version of the TOE provided for evaluation is uniquely referenced.

1299 The evaluator should use the developer's CM system to validate the uniqueness of the reference by checking the configuration list to ensure that the configuration items are uniquely identified. Evidence that the version provided for evaluation is uniquely referenced may be incomplete if only one version is examined during the evaluation, and the evaluator should look for a referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates). However, the absence of any reference will normally lead to a fail verdict against this requirement unless the evaluator is confident that the TOE can be uniquely identified.

1300 The evaluator should seek to examine more than one version of the TOE (e.g. during rework following discovery of a vulnerability), to check that the two versions are referenced differently.

ACM_CAP.4.2C

4:ACM_CAP.4-2 The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

1301 The evaluator should ensure that the TOE contains a unique reference such that it is possible to distinguish different versions of the TOE. This could be achieved through labelled packaging or media, or by a label displayed by the operational TOE. This is to ensure that it would be possible for consumers to identify the TOE (e.g. at the point of purchase or use).

1302 The TOE may provide a method by which it can be easily identified. For example, a software TOE may display its name and version number during the start up routine, or in response to a command line entry. A hardware or firmware TOE may be identified by a part number physically stamped on the TOE.

4:ACM_CAP.4-3 The evaluator ***shall check*** that the TOE references used are consistent.

1303 If the TOE is labelled more than once then the labels have to be consistent. For example, it should be possible to relate any labelled guidance documentation supplied as part of the TOE to the evaluated operational TOE. This ensures that consumers can be confident that they have purchased the evaluated version of the TOE, that they have installed this version, and that they have the correct version of the guidance to operate the TOE in accordance with its ST. The evaluator can use the configuration list that is part of the provided CM documentation to verify the consistent use of identifiers.

1304 The evaluator also verifies that the TOE reference is consistent with the ST.

1305 For guidance on consistency analysis see Annex B.3.

ACM_CAP.4.3C

4:ACM_CAP.4-4 The evaluator ***shall check*** that the CM documentation provided includes a configuration list.

1306 A configuration list identifies the items being maintained under configuration control.

4:ACM_CAP.4-5 The evaluator ***shall check*** that the CM documentation provided includes a CM plan.

4:ACM_CAP.4-6 The evaluator ***shall check*** that the CM documentation provided includes an acceptance plan.

Interp Note : The following element is added as a result of Interpretation 003.

ACM_CAP.2.<new C&P element resulting from Interp>C

4:ACM_CAP.2-new The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.

The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.

ACM_CAP.4.4C

4:ACM_CAP.4-7 The evaluator ***shall examine*** the configuration list to determine that it identifies the configuration items that comprise the TOE.

1307 The minimum scope of configuration items to be covered in the configuration list is given by ACM_SCP.

ACM_CAP.4.5C

4:ACM_CAP.4-8 The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

ACM_CAP.4.6C

Interp Note : The following element is replaced as a result of Interpretation 003.

4:ACM_CAP.4-9 ~~The evaluator ***shall check*** that the configuration list uniquely identifies each configuration item.~~ The evaluator shall examine the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

1308 ~~The configuration list contains a list of the configuration items that comprise the TOE, together with sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.~~Assurance that the CM system uniquely identifies all configuration items is gained by examining the identifiers for the configuration items. For both configuration items that comprise the TOE, and drafts of configuration items that are submitted by the developer as evaluation evidence, the evaluator confirms that each configuration item possesses a unique identifier in a manner consistent with the unique identification method that is described in the CM documentation.

ACM_CAP.4.7C

4:ACM_CAP.4-10 The evaluator ***shall examine*** the CM plan to determine that it describes how the CM system is used to maintain the integrity of the TOE configuration items.

1309 The descriptions contained in a CM plan may include:

a) all activities performed in the TOE development environment that are subject to configuration management procedures (e.g. creation, modification or deletion of a configuration item);

b) the roles and responsibilities of individuals required to perform operations on individual configuration items (different roles may be identified for different types of configuration item (e.g. design documentation or source code));

c) the procedures that are used to ensure that only authorised individuals can make changes to configuration items;

d) the procedures that are used to ensure that concurrency problems do not occur as a result of simultaneous changes to configuration items;

e) the evidence that is generated as a result of application of the procedures. For example, for a change to a configuration item, the CM system might record a description of the change, accountability for the change, identification of all configuration items affected, status (e.g. pending or completed), and date and time of the change. This might be recorded in an audit trail of changes made or change control records;

f) the approach to version control and unique referencing of TOE versions (e.g. covering the release of patches in operating systems, and the subsequent detection of their application).

ACM_CAP.4.8C

4:ACM_CAP.4-11 The evaluator ***shall check*** the CM documentation to ascertain that it includes the CM system records identified by the CM plan.

1310 The output produced by the CM system should provide the evidence that the evaluator needs to be confident that the CM plan is being applied, and also that all configuration items are being maintained by the CM system as required by ACM_CAP.4.9C. Example output could include change control forms, or configuration item access approval forms.

4:ACM_CAP.4-12 The evaluator ***shall examine*** the evidence to determine that the CM system is being used as it is described in the CM plan.

1311 The evaluator should select and examine a sample of evidence covering each type of CM-relevant operation that has been performed on a configuration item (e.g. creation, modification, deletion, reversion to an earlier version) to confirm that all operations of the CM system have been carried out in line with documented procedures. The evaluator confirms that the evidence includes all the information identified for that operation in the CM plan. Examination of the evidence may require access to a CM tool that is used. The evaluator may choose to sample the evidence.

1312 For guidance on sampling see Annex B.2.

1313 Further confidence in the correct operation of the CM system and the effective maintenance of configuration items may be established by means of interview with selected development staff. In conducting such interviews, the evaluator should aim to gain a deeper understanding of how the CM system is used in practice as well as to confirm that the CM procedures are being applied as described in the

CM documentation. Note that such interviews should complement rather than replace the examination of documentary evidence, and may not be necessary if the documentary evidence alone satisfies the requirement. However, given the wide scope of the CM plan it is possible that some aspects (e.g. roles and responsibilities) may not be clear from the CM plan and records alone. This is one case where clarification may be necessary through interviews.

1314 It is expected that the evaluator will visit the development site in support of this activity.

1315 For guidance on site visits see Annex B.5.

ACM_CAP.4.9C

4:ACM_CAP.4-13 The evaluator ***shall check*** that the configuration items identified in the configuration list are being maintained by the CM system.

1316 The CM system employed by the developer should maintain the integrity of the TOE. The evaluator should check that for each type of configuration item (e.g. high-level design or source code modules) contained in the configuration list there are examples of the evidence generated by the procedures described in the CM plan. In this case, the approach to sampling will depend upon the level of granularity used in the CM system to control CM items. Where, for example, 10,000 source code modules are identified in the configuration list, a different sampling strategy should be applied compared to the case in which there are only 5, or even 1. The emphasis of this activity should be on ensuring that the CM system is being operated correctly, rather than on the detection of any minor error.

1317 For guidance on sampling see Annex B.2.

ACM_CAP.4.10C

4:ACM_CAP.4-14 The evaluator ***shall examine*** the CM access control measures described in the CM plan to determine that they are effective in preventing unauthorised access to the configuration items.

1318 The evaluator may use a number of methods to determine that the CM access control measures are effective. For example, the evaluator may exercise the access control measures to ensure that the procedures could not be bypassed. The evaluator may use the outputs generated by the CM system procedures and already examined as part of the work unit 4:ACM_CAP.4-13. The evaluator may also witness a demonstration of the CM system to ensure that the access control measures employed are operating effectively.

1319 The developer will have provided automated access control measures as part of the CM system and as such their suitability may be verified under the component ACM_AUT.1

ACM_CAP.4.11C

4:ACM_CAP.4-15 The evaluator ***shall check*** the CM documentation for procedures for supporting the generation of the TOE.

1320 In this work unit the term *generation* applies to those processes adopted by the developer to progress the TOE from implementation to a state acceptable for delivery to the end customer.

1321 The evaluator verifies the existence of generation support procedures within the CM documentation. The generation support procedures provided by the developer may be automated, and as such their existence may be verified under the component ACM_AUT.1.2C.

4:ACM_CAP.4-16 The evaluator ***shall examine*** the TOE generation procedures to determine that they are effective in helping to ensure that the correct configuration items are used to generate the TOE.

1322 The evaluator determines that by following the generation support procedures the version of the TOE expected by the customer (i.e. as described in the TOE ST and consisting of the correct configuration items) would be generated and delivered for installation at the customer site. For example, in a software TOE this may include checking that the procedures ensure that all source files and related libraries are included in the compiled object code.

1323 The evaluator should bear in mind that the CM system need not necessarily possess the capability to generate the TOE, but should provide support for the process that will help reduce the probability of human error.

ACM_CAP.4.12C

4:ACM_CAP.4-17 The evaluator ***shall examine*** the acceptance procedures to determine that they describe the acceptance criteria to be applied to newly created or modified configuration items.

1324 An acceptance plan describes the procedures that are to be used to ensure that the constituent parts of the TOE are of adequate quality prior to incorporation into the TOE. The acceptance plan should identify the acceptance procedures to be applied:

a) at each stage of the construction of the TOE (e.g. module, integration, system);

b) to the acceptance of software, firmware and hardware components;

c) to the acceptance of previously evaluated components.

1325 The description of the acceptance criteria may include identification of:

a) developer roles or individuals responsible for accepting such configuration items;

b) any acceptance criteria to be applied before the configuration items are accepted (e.g. successful document review, or successful testing in the case of software, firmware or hardware).

8.4.3 Evaluation of CM scope (ACM_SCP.2)

8.4.3.1 Objectives

Interp Note : The following paragraph is changed as a result of Interpretation 038.

1326 The objective of this sub-activity is to determine whether ~~as a minimum~~ the developer performs configuration management on the TOE implementation representation, design, tests, user and administrator guidance, the CM documentation and security flaws.

8.4.3.2 Input

Interp Note : The following paragraph is changed as a result of Interpretation 004.

1327 The evaluation evidence for this sub-activity is:

a) the configuration ~~management documentation~~ item list.

8.4.3.3 Evaluator action

1328 This sub-activity comprises one CC Part 3 evaluator action element:

a) ACM_SCP.2.1E.

8.4.3.3.1 Action ACM_SCP.2.1E

ACM_SCP.2.1C

Interp Note : The following element is changed as a result of Interpretation 004.

4:ACM_SCP.2-1 The evaluator ***shall check*** that the configuration item list includes the ~~minimum~~ set of items required by the CC ~~to be tracked by the CM system~~.

Interp Note : The following paragraph is changed as a result of Interpretations 004 and 038.

1329 The list should include at least the following ~~as a minimum~~:

~~a) all documentation required to meet the target level of assurance;~~

~~b) other design documentation (e.g. low-level design);~~

~~c) test software (if applicable);~~

~~d) the TOE implementation representation (i.e. the components or subsystems that compose the TOE). For a software-only TOE, the implementation representation may consist solely of source code; for a TOE that includes a hardware platform, the implementation representation may refer to a combination of software, firmware and a description of the hardware (or a reference platform).~~

a) the TOE implementation representation (i.e. the components or subsystems that compose the TOE). For a software-only TOE, the implementation representation may consist solely of source code; for a TOE that includes a hardware platform, the implementation representation may refer to a combination of software, firmware and a description of the hardware;

b) the evaluation evidence documentation required to by the assurance components in the ST.

c) the documentation used to record details of reported security flaws associated with the implementation (e.g. problem status reports derived from a developer's problem reporting database).

Interp Note : The following work unit is deleted as a result of Interpretation 004.

~~ACM_SCP.2.2C~~

4:ACM_SCP.2-2 ~~The evaluator ***shall examine*** the CM documentation to determine that the procedures describe how the status of each configuration item can be tracked throughout the lifecycle of the TOE.~~

1330 ~~The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:~~

~~a) how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;~~

~~b) how configuration items are assigned unique identifiers and how they are entered into the CM system;~~

~~c) the method to be used to identify superseded versions of a configuration item;~~

~~d) the method to be used for identifying and tracking configuration items through each stage of the TOE development and maintenance lifecycle (i.e. requirements specification, design, source code development, through to object code generation and on to executable code, module testing, implementation and operation);~~

~~e) the method used for assigning the current status of the configuration item at a given point in time and for tracking each configuration item through the various levels of representation at the development phase (i.e. source code development, through to object code generation and on to executable code, module testing and documentation);~~

~~f) the method used for identifying and tracking flaws relative to configuration items throughout the development lifecycle;~~

g) ~~the method used for identifying correspondence between configuration items such that if one configuration item is changed it can be determined which other configuration items will also need to be changed.~~

1331 ~~The analysis of the CM documentation for some of this information may have been satisfied by work units detailed under ACM_CAP.~~

8.5 Delivery and operation activity

1332 The purpose of the delivery and operation activity is to judge the adequacy of the documentation of the procedures used to ensure that the TOE is installed, generated, and started in the same way the developer intended it to be and that it is delivered without modification. This includes both the procedures taken while the TOE is in transit, as well as the initialisation, generation, and start-up procedures.

1333 The delivery and operation activity at EAL4 contains sub-activities related to the following components:

a) ADO_DEL.2;

b) ADO_IGS.1.

8.5.1 Evaluation of delivery (ADO_DEL.2)

8.5.1.1 Objectives

Interp Note : The following paragraph is changed as a result of Interpretation 016.

1334 The objective of this sub-activity is to determine whether the delivery documentation describes all procedures used to maintain ~~integrity~~ security and ~~the detection of~~ detect modification or substitution of the TOE when distributing the TOE to the user's site.

8.5.1.2 Input

1335 The evaluation evidence for this sub-activity is:

a) the delivery documentation.

8.5.1.3 Evaluator actions

1336 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_DEL.2.1E;

b) implied evaluator action based on ADO_DEL.2.2D.

8.5.1.3.1 Action ADO_DEL.2.1E

ADO_DEL.2.1C

4:ADO_DEL.2-1 The evaluator ***shall examine*** the delivery documentation to determine that it describes all procedures that are necessary to maintain security when distributing versions of the TOE or parts of it to the user's site.

1337 Interpretation of the term *necessary* will need to consider the nature of the TOE and information contained in the ST. The level of protection provided should be commensurate with the assumptions, threats, organisational security policies, and security objectives identified in the ST. In some cases these may not be explicitly expressed in relation to delivery. The evaluator should determine that a balanced approach has been taken, such that delivery does not present an obvious weak point in an otherwise secure development process.

Interp Note : The following paragraph is changed as a result of Interpretation 128, rev 1.

1338 ~~The delivery procedures describe proper procedures to determine the identification of the TOE and to maintain integrity during transfer of the TOE or its component parts. The procedures describe which parts of the TOE need to be covered by these procedures. It should contain procedures for physical or electronic (e.g. for downloading off the Internet) distribution where applicable. The delivery procedures refer to the entire TOE, including applicable software, hardware, firmware and documentation.~~

The delivery documentation describes proper procedures to determine the identification of the TOE and to maintain security of the TOE during transfer of the TOE or its component parts. The delivery documentation contains procedures for physical or electronic (e.g. for downloading off the Internet) distribution where applicable. The delivery documentation covers the entire TOE, but may contain different procedures for different parts of the TOE.

Interp Note : The following paragraph is changed as a result of Interpretation 016.

1339 ~~The emphasis on integrity is not surprising, since integrity will always be of concern for TOE delivery. Where confidentiality and availability are of concern, they also should be considered under this work unit.~~ The emphasis in the delivery documentation is likely to be on measures related to integrity, as technical measures are required to be applied to maintain integrity during the TOE delivery. However, confidentiality and availability of the delivery will be of concern in the delivery of some TOEs; procedures relating to these aspects of the secure delivery should also be discussed in the procedures.

1340 The delivery procedures should be applicable across all phases of delivery from the production environment to the installation environment (e.g. packaging, storage and distribution).

Interp Note : The following two paragraphs are added as a result of Interpretation 116.

Standard commercial practice for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution, the public mail or a private distribution service may be acceptable.

The suitability of the choice of the delivery procedures is influenced by the TOE (e.g. whether it is software or hardware) and by the security objectives. In cases where the delivery procedures differ for different parts of the TOE, the totality of procedures are suitable to meet the overall security objectives.

Interp Note : The following work unit and its two guidance are deleted as a result of Interpretation 116.

4:ADO_DEL.2-2 ~~The evaluator ***shall examine*** the delivery procedures to determine that the chosen procedure and the part of the TOE it covers is suitable to meet the security objectives.~~

1341 ~~The suitability of the choice of the delivery procedures is influenced by the specific TOE (e.g. whether it is software or hardware) and by the security objectives.~~

1342 ~~Standard commercial practice for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution the public mail or a private distribution service may be acceptable.~~

ADO_DEL.2.2C

4:ADO_DEL.2-3 The evaluator ***shall examine*** the delivery documentation to determine that it describes how the various procedures and technical measures provide for the detection of modifications or any discrepancy between the developer's master copy and the version received at the user site.

1343 Checksum procedures, software signature, or tamper proof seals may be used by the developer to ensure that tampering can be detected. The developer may also employ other procedures (e.g. a recorded delivery service) that register the name of the originator and supply the name to the receiver.

1344 Technical measures for the detection of any discrepancy between the developer's master copy and the version received at the user site should be described in the delivery procedures.

ADO_DEL.2.3C

4:ADO_DEL.2-4 The evaluator ***shall examine*** the delivery documentation to determine that it describes how the various mechanisms and procedures allow detection of attempted masquerading even in cases in which the developer has sent nothing to the user's site.

1345 This requirement may be fulfilled by delivering the TOE or parts of it (e.g. by an agent known to and trusted by both developer and user). For a software TOE a digital signature may be appropriate.

1346 If the TOE is delivered by electronic download, the security can be maintained by using digital signatures, integrity checksums, or encryption.

8.5.1.3.2 Implied evaluator action

ADO_DEL.2.2D

4:ADO_DEL.2-5 The evaluator ***shall examine*** aspects of the delivery process to determine that the delivery procedures are used.

1347 The approach taken by the evaluator to check the application of delivery procedures will depend on the nature of the TOE, and the delivery process itself. In addition to examination of the procedures themselves, the evaluator should seek some assurance that they are applied in practice. Some possible approaches are:

a) a visit to the distribution site(s) where practical application of the procedures may be observed;

b) examination of the TOE at some stage during delivery, or at the user's site (e.g. checking for tamper proof seals);

c) observing that the process is applied in practice when the evaluator obtains the TOE through regular channels;

d) questioning end users as to how the TOE was delivered.

1348 For guidance on site visits see Annex B.5.

1349 It may be the case of a newly developed TOE that the delivery procedures have yet to be exercised. In these cases, the evaluator has to be satisfied that appropriate procedures and facilities are in place for future deliveries and that all personnel involved are aware of their responsibilities. The evaluator may request a "dry run" of a delivery if this is practical. If the developer has produced other similar products, then an examination of procedures in their use may be useful in providing assurance.

8.5.2 Evaluation of installation, generation and start-up (ADO_IGS.1)

8.5.2.1 Objectives

1350 The objective of this sub-activity is to determine whether the procedures and steps for the secure installation, generation, and start-up of the TOE have been documented and result in a secure configuration.

8.5.2.2 Input

1351 The evaluation evidence for this sub-activity is:

a) the administrator guidance;

b) the secure installation, generation, and start-up procedures;

c) the TOE suitable for testing.

8.5.2.3 Application notes

1352 The installation, generation, and start-up procedures refer to all installation, generation, and start-up procedures, regardless of whether they are performed at the user's site or at the development site that are necessary to progress the TOE to the secure configuration as described in the ST.

8.5.2.4 Evaluator actions

1353 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADO_IGS.1.1E;

b) ADO_IGS.1.2E.

8.5.2.4.1 Action ADO_IGS.1.1E

ADO_IGS.1.1C

4:ADO_IGS.1-1 The evaluator ***shall check*** that the procedures necessary for the secure installation, generation and start-up of the TOE have been provided.

1354 If it is not anticipated that the installation, generation, and start-up procedures will or can be reapplied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

8.5.2.4.2 Action ADO_IGS.1.2E

4:ADO_IGS.1-2 The evaluator ***shall examine*** the provided installation, generation, and start-up procedures to determine that they describe the steps necessary for secure installation, generation, and start-up of the TOE.

1355 If it is not anticipated that the installation, generation, and start-up procedures will or can be reapplied (e.g. because the TOE may already be delivered in an operational state) this work unit (or the effected parts of it) is not applicable, and is therefore considered to be satisfied.

1356 The installation, generation, and start-up procedures may provide detailed information about the following:

a) changing the installation specific security characteristics of entities under the control of the TSF;

b) handling exceptions and problems;

c) minimum system requirements for secure installation if applicable.

1357 In order to confirm that the installation, generation, and start-up procedures result in a secure configuration, the evaluator may follow the developer's procedures and may perform the activities that customers are usually expected to perform to install, generate, and start-up the TOE (if applicable to the TOE), using the supplied guidance documentation only. This work unit might be performed in conjunction with the 4:ATE_IND.2-2 work unit.

8.6 Development activity

1358 The purpose of the development activity is to assess the design documentation in terms of its adequacy to understand how the TSF provides the security functions of the TOE. This understanding is achieved through examination of increasingly refined descriptions of the TSF design documentation. Design documentation consists of a functional specification (which describes the external interfaces of the TOE), a high-level design (which describes the architecture of the TOE in terms of internal subsystems), and a low-level design (which describes the architecture of the TOE in terms of internal modules). Additionally, there is an implementation description (a source code level description), a security policy model (which describes the security policies enforced by the TOE) and a representation correspondence (which maps representations of the TOE to one another in order to ensure consistency).

1359 The development activity at EAL4 contains sub-activities related to the following components:

a) ADV_FSP.2;

b) ADV_HLD.2;

c) ADV_IMP.1;

d) ADV_LLD.1;

e) ADV_RCR.1;

f) ADV_SPM.1.

8.6.1 Application notes

1360 The CC requirements for design documentation are levelled by formality. The CC considers a document's degree of formality (that is, whether it is informal, semiformal or formal) to be hierarchical. An informal document is one that is expressed in a natural language. The methodology does not dictate the specific language that must be used; that issue is left for the scheme. The following paragraphs differentiate the contents of the different informal documents.

1361 An informal functional specification comprises a description the security functions (at a level similar to that of the TOE summary specification) and a description of the externally-visible interfaces to the TSF. For example, if an operating system presents the user with a means of self-identification, of creating files, of modifying or deleting files, of setting permissions defining what other users may access files, and of communicating with remote machines, its functional specification would contain descriptions of each of these functions. If there are also audit functions that detect and record the occurrences of such events, descriptions of these audit functions would also be expected to be part of the functional specification; while these functions are technically not directly invoked by the user at the external interface, they certainly are affected by what occurs at the user's external interface.

1362 An informal high-level design is expressed in terms of sequences of actions that occur in each subsystem in response to stimulus at its interface. For example, a firewall might be composed of subsystems that deal with packet filtering, with remote administration, with auditing, and with connection-level filtering. The high-level design description of the firewall would describe the actions that are taken, in terms of what actions each subsystem takes when an incoming packet arrives at the firewall.

1363 An informal low-level design is expressed in terms of sequences of actions that occur in a module in response to stimulus at its interface. For example, a virtual private networking subsystem might be composed of modules that create session keys, that encrypt traffic, that decrypt traffic, and that decide whether traffic needs to be encrypted. The low-level description of the encryption module would describe the steps that the module takes when it receives a traffic stream that is to be encrypted.

1364 While the functional specification describes the functions and services, the model describes the policies those functions and services enforce. An informal model is simply a description of the security policies enforced by services or functions available at the external interface. For example, access control policies would describe the resources being protected and the conditions that must be met for access to be granted; audit policies would describe the TOE's auditable events, identifying both those that are selectable by the administrator and those that are always audited; identification and authentication policies would describe how users are identified, how those claimed identities are authenticated, and any rules affecting how identities are authenticated (e.g. users on the corporate intranet need no authentication, while external users are authenticated with one-time passwords).

1365 Informality of the demonstration of correspondence need not be in a prose form; a simple two-dimensional mapping may be sufficient. For example, a matrix with modules listed along one axis and subsystems listed along the other, with the cells identifying the correspondence of the two, would serve to provide an adequate informal correspondence between the high-level design and the low-level design.

8.6.2 Evaluation of functional specification (ADV_FSP.2)

8.6.2.1 Objectives

1366 The objective of this sub-activity is to determine whether the developer has provided an adequate description of all security functions of the TOE and whether the security functions provided by the TOE are sufficient to satisfy the security functional requirements of the ST.

8.6.2.2 Input

1367 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance.

8.6.2.3 Evaluator actions

1368 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_FSP.2.1E;

b) ADV_FSP.2.2E.

8.6.2.3.1 Action ADV_FSP.2.1E

ADV_FSP.2.1C

4:ADV_FSP.2-1 The evaluator ***shall examine*** the functional specification to determine that it contains all necessary informal explanatory text.

1369 If the entire functional specification is informal, this work unit is not applicable and is therefore considered to be satisfied.

1370 Supporting narrative descriptions are necessary for those portions of the functional specification that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_FSP.2.2C

4:ADV_FSP.2-2 The evaluator ***shall examine*** the functional specification to determine that it is internally consistent.

1371 The evaluator validates the functional specification by ensuring that the descriptions of the interfaces making up the TSFI are consistent with the descriptions of the functions of the TSF.

1372 For guidance on consistency analysis see Annex B.3.

ADV_FSP.2.3C

4:ADV_FSP.2-3 The evaluator ***shall examine*** the functional specification to determine that it identifies all of the external TOE security function interfaces.

1373 The term *external* refers to that which is visible to the user. External interfaces to the TOE are either direct interfaces to the TSF or interfaces to non-TSF portions of the TOE. However, these non-TSF interfaces might have eventual access to the TSF. These external interfaces that directly or indirectly access the TSF collectively make up the TOE security function interface (TSFI). Figure 8.1 shows a TOE with TSF (shaded) portions and non-TSF (empty) portions. This TOE has three external interfaces: interface *c* is a direct interface to the TSF; interface *b* is an indirect interface to the TSF; and interface *a* is an interface to non-TSF portions of the TOE. Therefore, interfaces *b* and *c* make up the TFSI.

1374 It should be noted that all security functions reflected in the functional requirements of CC Part 2 (or in extended components thereof) will have some sort of externally-visible manifestation. While not all of these are necessarily interfaces from which the security function can be tested, they are all externally-visible to some extent and must therefore be included in the functional specification.

1375 For guidance on determining the TOE boundary see Annex B.6.

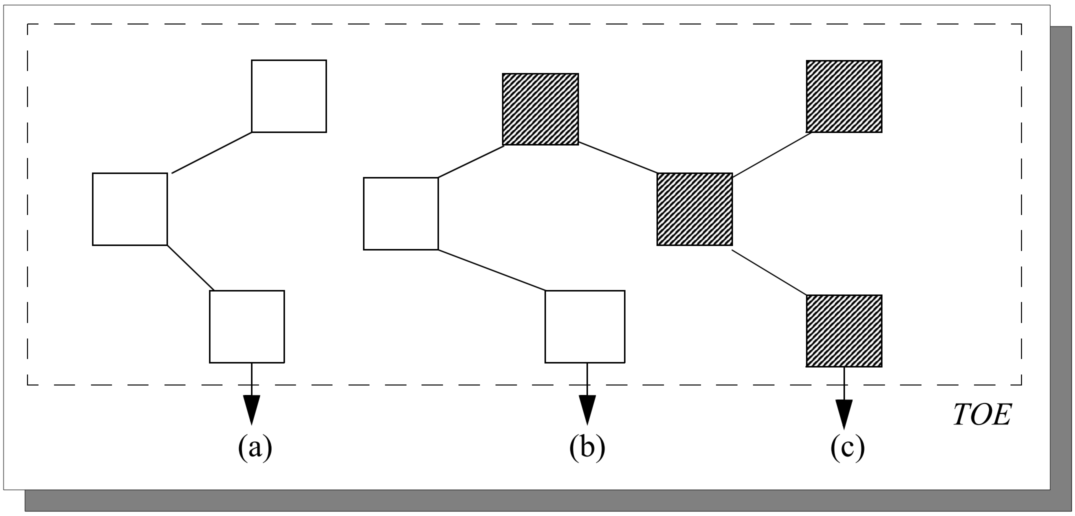

Figure 8.1 TSF Interfaces

4:ADV_FSP.2-4 The evaluator ***shall examine*** the functional specification to determine that it describes all of the external TOE security function interfaces.

1376 For a TOE that has no threat of malicious users (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are rightfully excluded from its ST), the only interfaces that are described in the functional specification (and expanded upon in the other TSF representation descriptions) are those to and from the TSF. The absence of FPT_PHP, FPT_RVM, and FPT_SEP presumes there is no concern for any sort of bypassing of the security features; therefore, there is no concern with any possible impact that other interfaces might have on the TSF.

1377 On the other hand, if the TOE has a threat of malicious users or bypass (i.e. FPT_PHP, FPT_RVM, and FPT_SEP are included in its ST), all external interfaces are described in the functional specification, but only to the extent that the effect of each is made clear: interfaces to the security functions (i.e. interfaces *b* and *c* in Figure 8.1) are completely described, while other interfaces are described only to the extent that it is clear that the TSF is inaccessible through the interface (i.e. that the interface is of type *a*, rather than *b* in Figure 8.1). The inclusion of FPT_PHP, FPT_RVM, and FPT_SEP implies a concern that all interfaces might have some effect upon the TSF. Because each external interface is a potential TSF interface, the functional specification must contain a description of each interface in sufficient detail so that an evaluator can determine whether the interface is security relevant.

1378 Some architectures lend themselves to readily provide this interface description in sufficient detail for groups of external interfaces. For example, a kernel architecture is such that all calls to the operating system are handled by kernel programs; any calls that might violate the TSP must be called by a program with the privilege to do so. All programs that execute with privilege must be included in the functional specification. Any program external to the kernel that executes without privilege is incapable of affecting the TSP (i.e. such programs are interfaces of type *a*, rather than *b* in Figure 8.1) and may, therefore, be excluded from the functional specification. It is worth noting that, while the evaluator's understanding of the interface description can be expedited in cases where there is a kernel architecture, such an architecture is not necessary.

4:ADV_FSP.2-5 The evaluator ***shall examine*** the presentation of the TSFI to determine that it adequately and correctly describes the complete behaviour of the TOE at each external interface describing effects, exceptions and error messages.

1379 In order to assess the adequacy and correctness of an interface's presentation, the evaluator uses the functional specification, the TOE summary specification of the ST, and the user and administrator guidance to assess the following factors:

a) All security relevant user input parameters (or a characterisation of those parameters) should be identified. For completeness, parameters outside of direct user control should be identified if they are usable by administrators.

b) Complete security relevant behaviour described in the reviewed guidance should be reflected in the description of semantics in the functional specification. This should include an identification of the behaviour in terms of events and the effect of each event. For example, if an operating system provides a rich file system interface, where it provides a different error code for each reason why a file is not opened upon request, the functional specification should explain that a file is either opened upon request, or else that the request is denied, along with a listing of the reasons why the open request might be denied (e.g. access denied, no such file, file is in use by another user, user is not authorised to open the file after 5pm, etc.). It would be insufficient for the functional specification merely to explain that a file is either opened upon request, or else that an error code is returned. The description of the semantics should include how the security requirements apply to the interface (e.g. whether the use of the interface is an auditable event and, if so, the information that can be recorded).

c) All interfaces are described for all possible modes of operation. If the TSF provides the notion of privilege, the description of the interface should explain how the interface behaves in the presence or absence of privilege.

d) The information contained in the descriptions of the security relevant parameters and syntax of the interface should be consistent across all documentation.

1380 Verification of the above is done by reviewing the functional specification and the TOE summary specification of the ST, as well as the user and administrator guidance provided by the developer. For example, if the TOE were an operating system and its underlying hardware, the evaluator would look for discussions of user-accessible programs, descriptions of protocols used to direct the activities of programs, descriptions of user-accessible databases used to direct the activities of programs, and for user interfaces (e.g. commands, application program interfaces) as applicable to the TOE under evaluation; the evaluator would also ensure that the processor instruction set is described.

1381 This review might be iterative, such that the evaluator would not discover the functional specification to be incomplete until the design, source code, or other evidence is examined and found to contain parameters or error messages that have been omitted from the functional specification.

ADV_FSP.2.4C

4:ADV_FSP.2-6 The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

1382 In order to assess the completeness of the TSF representation, the evaluator consults the TOE summary specification of the ST, the user guidance, and the administrator guidance. None of these should describe security functions that are absent from the TSF presentation of the functional specification.

ADV_FSP.2.5C

4:ADV_FSP.2-7 The evaluator ***shall examine*** the functional specification to determine that it contains a convincing argument that the TSF is completely represented by the functional specification.

1383 The evaluator determines that there is a convincing argument that there are no interfaces of the TSFI that are missing from the functional specification. This may include a description of the procedure or methodology that the developer used to ensure that all external interfaces are covered. The argument would prove inadequate if, for example, the evaluator discovers commands, parameters, error messages, or other interfaces to the TSF in other evaluation evidence, yet absent from the functional specification.

8.6.2.0.1 Action ADV_FSP.2.2E

4:ADV_FSP.2-8 The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the TOE security functional requirements.

1384 To ensure that all ST security functional requirements are covered by the functional specification, the evaluator may construct a map between the TOE summary specification and the functional specification. Such a map might be already provided by the developer as evidence for meeting the correspondence (ADV_RCR.*) requirements, in which case the evaluator need only verify the completeness of this mapping, ensuring that all security functional requirements are mapped onto applicable TSFI presentations in the functional specification.

4:ADV_FSP.2-9 The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the TOE security functional requirements.

1385 For each interface to a security function with specific characteristics, the detailed information in the functional specification must be exactly as it is specified in the ST. For example, if the ST contains user authentication requirements that the password length must be eight characters, the TOE must have eight-character passwords; if the functional specification describes six-character fixed length passwords, the functional specification would not be an accurate instantiation of the requirements.

1386 For each interface in the functional specification that operates on a controlled resource, the evaluator determines whether it returns an error code that indicates a possible failure due to enforcement of one of the security requirements; if no error code is returned, the evaluator determines whether an error code should be returned. For example, an operating system might present an interface to OPEN a controlled object. The description of this interface may include an error code that indicates that access was not authorised to the object. If such an error code does not exist, the evaluator should confirm that this is appropriate (because, perhaps, access mediation is performed on READs and WRITEs, rather than on OPENs).

8.6.3 Evaluation of high-level design (ADV_HLD.2)

8.6.3.1 Objectives

1387 The objective of this sub-activity is to determine whether the high-level design provides a description of the TSF in terms of major structural units (i.e. subsystems), provides a description of the interfaces to these structural units, and is a correct realisation of the functional specification.

8.6.3.2 Input

1388 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design.

8.6.3.3 Evaluator actions

1389 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_HLD.2.1E;

b) ADV_HLD.2.2E.

8.6.3.3.1 Action ADV_HLD.2.1E

ADV_HLD.2.1C

4:ADV_HLD.2-1 The evaluator ***shall examine*** the high-level design to determine that it contains all necessary informal explanatory text.

1390 If the entire high-level design is informal, this work unit is not applicable and is therefore considered to be satisfied.

1391 Supporting narrative descriptions are necessary for those portions of the high-level design that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_HLD.2.2C

4:ADV_HLD.2-2 The evaluator ***shall examine*** the presentation of the high-level design to determine that it is internally consistent.

1392 For guidance on consistency analysis see Annex B.3.

1393 The evaluator validates the subsystem interface specifications by ensuring that the interface specifications are consistent with the description of the purpose of the subsystem.

ADV_HLD.2.3C

4:ADV_HLD.2-3 The evaluator ***shall examine*** the high-level design to determine that the TSF is described in terms of subsystems.

1394 With respect to the high-level design, the term *subsystem* refers to large, related units (such as memory-management, file-management, process-management). Breaking a design into the basic functional areas aids in the understanding of the design.

1395 The primary purpose for examining the high-level design is to aid the evaluator's understanding of the TOE. The developer's choice of subsystem definition, and of the grouping of TSFs within each subsystem, are an important aspect of making the high-level design useful in understanding the TOE's intended operation. As part of this work unit, the evaluator should make an assessment as to the appropriateness of the number of subsystems presented by the developer, and also of the choice of grouping of functions within subsystems. The evaluator should ensure that the decomposition of the TSF into subsystems is sufficient for the evaluator to gain a high-level understanding of how the functionality of the TSF is provided.

1396 The subsystems used to describe the high-level design need not be called "subsystems", but should represent a similar level of decomposition. For example, the design may be decomposed using "layers" or "managers".

1397 There may be some interaction between the choice of subsystem definition and the scope of the evaluator's analysis. A discussion on this interaction is found following work unit 4:ADV_HLD.2-10.

ADV_HLD.2.4C

4:ADV_HLD.2-4 The evaluator ***shall examine*** the high-level design to determine that it describes the security functionality of each subsystem.

1398 The security functional behaviour of a subsystem is a description of what the subsystem does. This should include a description of any actions that the subsystem may be directed to perform through its functions and the effects the subsystem may have on the security state of the TOE (e.g. changes in subjects, objects, security databases).

ADV_HLD.2.5C

4:ADV_HLD.2-5 The evaluator ***shall check*** the high-level design to determine that it identifies all hardware, firmware, and software required by the TSF.

1399 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

1400 If the ST contains the optional statement of security requirements for the IT environment, the evaluator compares the list of hardware, firmware, or software required by the TSF as stated in the high-level design to the statement of security requirements for the IT environment to determine that they agree. The information in the ST characterises the underlying abstract machine on which the TOE will execute.

1401 If the high-level design includes security requirements for the IT environment that are not included in the ST, or if they differ from those included in the ST, this inconsistency is assessed by the evaluator under Action ADV_HLD.2.2E.

4:ADV_HLD.2-6 The evaluator ***shall examine*** the high-level design to determine that it includes a presentation of the functions provided by the supporting protection mechanisms implemented in the underlying hardware, firmware, or software.

1402 If the ST contains no security requirements for the IT environment, this work unit is not applicable and is therefore considered to be satisfied.

1403 The presentation of the functions provided by the underlying abstract machine on which the TOE executes need not be at the same level of detail as the presentation of functions that are part of the TSF. The presentation should explain how the TOE uses the functions provided in the hardware, firmware, or software that implement the security requirements for the IT environment that the TOE is dependent upon to support the TOE security objectives.

1404 The statement of security requirements for the IT environment may be abstract, particularly if it is intended to be capable of being satisfied by a variety of different combinations of hardware, firmware, or software. As part of the Tests activity, where the evaluator is provided with at least one instance of an underlying machine that is claimed to satisfy the security requirements for the IT environment, the evaluator can determine whether it provides the necessary security functions for the TOE. This determination by the evaluator does not require testing or analysis of the underlying machine; it is only a determination that the functions expected to be provided by it actually exist.

ADV_HLD.2.6C

4:ADV_HLD.2-7 The evaluator ***shall check*** that the high-level design identifies the interfaces to the TSF subsystems.

1405 The high-level design includes, for each subsystem, the name of each of its entry points.

ADV_HLD.2.7C

4:ADV_HLD.2-8 The evaluator ***shall check*** that the high-level design identifies which of the interfaces to the subsystems of the TSF are externally visible.

1406 As discussed under work unit 4:ADV_FSP.2-3, external interfaces (i.e. those visible to the user) may directly or indirectly access the TSF. Any external interface that accesses the TSF either directly or indirectly is included in the identification for this work unit. External interfaces that do not access the TSF need not be included.

ADV_HLD.2.8C

4:ADV_HLD.2-9 The evaluator ***shall examine*** the high-level design to determine that it describes the interfaces to each subsystem in terms of their purpose and method of use, and provides details of effects, exceptions and error messages, as appropriate.

1407 The high-level design should include descriptions in terms of the purpose and method of use for all interfaces of each subsystem. Such descriptions may be provided in general terms for some interfaces, and in more detail for others. In determining the level of detail of effects, exceptions and error messages that should be provided, the evaluator should consider the purposes of this analysis and the uses made of the interface by the TOE. For example, the evaluator needs to understand the nature of the interactions between subsystems to establish confidence that the TOE design is sound, and may be able to obtain this understanding with only a general description of some of the interfaces between subsystems. In particular, internal subsystem entry points that are not called by any other subsystem would not normally require detailed descriptions.

1408 The level of detail may also depend on the testing approach adopted to meet the ATE_DPT requirement. For example, a different amount of detail may be needed for a testing approach that tests only through external interfaces than one that tests through both external and internal subsystem interfaces.

1409 Detailed descriptions would include details of any input and output parameters, of the effects of the interface, and of any exceptions or error messages it produces. In the case of external interfaces, the required description is probably included in the functional specification and can be referenced in the high-level design without replication.

ADV_HLD.2.9C

4:ADV_HLD.2-10 The evaluator ***shall check*** that the high-level design describes the separation of the TOE into TSP-enforcing and other subsystems.

1410 The TSF comprises all the parts of the TOE that have to be relied upon for enforcement of the TSP. Because the TSF includes both functions that directly enforce the TSP, and also those functions that, while not directly enforcing the TSP, contribute to the enforcement of the TSP in a more indirect manner, all TSP-enforcing subsystems are contained in the TSF. Subsystems that play no role in TSP enforcement are not part of the TSF. An entire subsystem is part of the TSF if any portion of it is.

1411 As explained under work unit 4:ADV_HLD.2-3, the developer's choice of subsystem definition, and of the grouping of TSFs within each subsystem, are important aspects of making the high-level design useful in understanding the TOE's intended operation. However, the choice of grouping of TSFs within subsystems also affects the scope of the TSF, because a subsystem with *any* function that directly or indirectly enforces the TSP is part of the TSF. While the goal of understandability is important, it is also helpful to limit the extent of the TSF so as to reduce the amount of analysis that is required. The two goals of understandability and scope reduction may sometimes work against each other. The evaluator should bear this in mind when assessing the choice of subsystem definition.

8.6.3.3.2 Action ADV_HLD.2.2E

4:ADV_HLD.2-11 The evaluator ***shall examine*** the high-level design to determine that it is an accurate instantiation of the TOE security functional requirements.

1412 The evaluator analyses the high-level design for each TOE security function to ensure that the function is accurately described. The evaluator also ensures that the function has no dependencies that are not included in the high-level design.

1413 The evaluator also analyses the security requirements for the IT environment in both the ST and the high-level design to ensure that they agree. For example, if the ST includes TOE security functional requirements for the storage of an audit trail, and the high-level design stated that audit trail storage is provided by the IT environment, then the high-level design is not an accurate instantiation of the TOE security functional requirements.

1414 The evaluator should validate the subsystem interface specifications by ensuring that the interface specifications are consistent with the description of the purpose of the subsystem.

4:ADV_HLD.2-12 The evaluator ***shall examine*** the high-level design to determine that it is a complete instantiation of the TOE security functional requirements.

1415 To ensure that all ST security functional requirements are covered by the high-level design, the evaluator may construct a map between the TOE security functional requirements and the high-level design.

8.6.4 Evaluation of implementation representation (ADV_IMP.1)

8.6.4.1 Objectives

1416 The objective of this sub-activity is to determine whether the implementation representation is sufficient to satisfy the functional requirements of the ST and is a correct realisation of the low-level design.

8.6.4.2 Input

1417 The evaluation evidence for this sub-activity is:

a) the ST;

b) the low-level design;

c) the subset of the implementation representation.

8.6.4.3 Evaluator actions

1418 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_IMP.1.1E;

b) ADV_IMP.1.2E.

8.6.4.3.1 Action ADV_IMP.1.1E

ADV_IMP.1.1C

4:ADV_IMP.1-1 The evaluator ***shall examine*** the implementation representation to determine that it unambiguously defines the TSF to a level of detail such that the TSF can be generated without any further design decisions.

1419 This work unit requires the evaluator to confirm that the implementation representation is suitable for analysis. The evaluator should consider the process needed to generate the TSF from the representation provided. If the process is well-defined, requiring no further design decisions (for example, requiring only the compilation of source code, or the building of hardware from hardware drawings), then the implementation representation can be said to be suitable.

1420 Any programming languages used must be well defined with an unambiguous definition of all statements, as well as the compiler options used to generate the object code. This determination will have been made as part of the ALC_TAT.1 sub-activity.

4:ADV_IMP.1-2 The evaluator ***shall examine*** the implementation representation provided by the developer to determine that it is sufficiently representative.

1421 The developer is required to provide the implementation representation for only a subset of the TSF. If the PP or ST specifies a selected subset, then the specified subset is also required of the developer. The developer can select and offer an initial subset, but the evaluator may require additional portions, or even different subsets.

1422 The evaluator determines the adequacy and appropriateness of the subset by applying the principles of sampling.

1423 For guidance on sampling see Annex B.2.

1424 In determining the appropriateness of the subset, the evaluator decides if it is suitable for use in aiding the evaluator to understand and gain assurance of the correctness of the implementation of the TSF mechanisms. In making this determination, the evaluator should consider the different methods of representation used by the developer, so that the evaluator is satisfied that a representative subset has been selected.

1425 For example, for a TOE that is realised in the manner of a conventional operating system, the selected subset of source code should include samples from the kernel or nucleus as well as samples from outside the kernel, such as command or application programs. If some of the source code is known to have originated from different development organisations, the selected subset should contain samples from each of the different creating organisations. If the implementation representation source code includes different forms of programming languages, the subset should contain samples of each different language.

1426 In the case that the implementation representation includes hardware drawings, several different portions of the TOE should be included in the subset. For example, for a TOE including a desktop computer, the selected subset should contain samples for peripheral controllers as well as the main computer board.

1427 Other factors that might influence the determination of the subset include:

a) the complexity of the design (if the design complexity varies across the TOE, the subset should include some portions with high complexity);

b) scheme requirements;

c) the results of other design analysis sub-activities (such as work units related to the low-level or high-level design) that might indicate portions of the TOE in which there is a potential for ambiguity in the design; and

d) the evaluator's judgement as to portions of the implementation representation that might be useful for the evaluator's independent vulnerability analysis (sub-activity AVA_VLA.2).

ADV_IMP.1.2C

4:ADV_IMP.1-3 The evaluator ***shall examine*** the implementation representation to determine that it is internally consistent.

1428 Because the developer is required to provide only a subset of the implementation representation, this work unit calls on the evaluator to make a determination of consistency only for the subset provided. The evaluator looks for inconsistencies by comparing portions of the implementation representation. In the case of source code, for example, if one portion of the source code includes a call to a subprogram in another portion, the evaluator looks to see that the arguments of the calling program match the called program's handling of the arguments. In the case of hardware drawings, the evaluator looks for such things as agreement between the nature and characteristics of the two ends of a circuit trace (e.g. voltage level, direction of logic, signal timing requirements). For guidance on consistency analysis see Annex B.3.

8.6.4.3.2 Action ADV_IMP.1.2E

4:ADV_IMP.1-4 The evaluator ***shall examine*** the implementation representation subset to determine that it accurately instantiates those TOE security functional requirements relevant to the subset.

1429 For those portions of the implementation representation subset that provide security functions directly, the evaluator determines that the implementation matches the TOE security functional requirement. The remaining portions of the implementation representation subset may support some TOE functional requirement. In making a determination about these remaining portions, the evaluator makes use of the low-level design to assess if the portions in the implementation representation subset, in combination with other portions as described in the low-level design, work together to instantiate a TOE security functional requirement.

1430 The remaining portions of the implementation representation subset, if any, can generally be ignored because they are unrelated to any of the TOE security functional requirements supported by the implementation subset. However, the evaluator should be careful to not overlook any portions that play an indirect role, no matter how distant, in supporting the TOE security functions. For example, in typical operating systems, the source code for portions of the nucleus (or kernel) may not have any direct role in supporting a TOE security function, but is capable of interfering with the correct functioning of those portions of the nucleus that *do* have a direct role. If any such portions are found to exist in the subset of the implementation representation provided, they should be assessed not to interfere with the portions that do, provided that the ST requires such non-interference. This assessment typically will not require the same level of detailed examination that is required for those portions of the implementation representation that play a more direct role in supporting the TOE security functions.

8.6.5 Evaluation of low-level design (ADV_LLD.1)

8.6.5.1 Objectives

1431 The objective of this sub-activity is to determine whether the low-level design is sufficient to satisfy the functional requirements of the ST, and is a correct and effective refinement of the high-level design.

8.6.5.2 Input

1432 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the low-level design.

8.6.5.3 Evaluator actions

1433 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ADV_LLD.1.1E;

b) ADV_LLD.1.2E.

8.6.5.3.1 Action ADV_LLD.1.1E

ADV_LLD.1.C

4:ADV_LLD.1-1 The evaluator ***shall examine*** the low-level design to determine that it contains all necessary informal explanatory text.

1434 If the entire low-level design is informal, this work unit is not applicable and is therefore considered to be satisfied.

1435 Supporting narrative descriptions are necessary for those portions of the low-level design that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_LLD.1.2C

4:ADV_LLD.1-2 The evaluator ***shall examine*** the presentation of the low-level design to determine that it is internally consistent.

1436 For guidance on consistency analysis see Annex B.3.

ADV_LLD.1.3C

4:ADV_LLD.1-3 The evaluator ***shall check*** the low-level design to determine that it describes the TSF in terms of modules.

1437 The term *module* is used in this family by the CC to denote a less abstract entity than a subsystem. This means that it contains more detail as to, not only the module's purpose, but also the manner in which the module achieves its purpose. Ideally, the low-level design would provide all the information needed to implement the modules described in it. The later work units in this sub-activity call for specific analysis to determine that a sufficient level of detail is included. For this work unit, it is sufficient for the evaluator to verify that each module is clearly and unambiguously identified.

ADV_LLD.1.4C

4:ADV_LLD.1-4 The evaluator ***shall examine*** the low-level design to determine that it describes the purpose of each module.

1438 The low-level design contains a description of the purpose of each of its modules. These descriptions should be clear enough to convey what functions the module is expected to perform. The description should provide an overview of a module's purpose and is not intended to be at the level of detail of module interface specifications.

ADV_LLD.1.5C

4:ADV_LLD.1-5 The evaluator ***shall examine*** the low-level design to determine that it defines the interrelationships between the modules in terms of provided security functionality and dependencies on other modules.

1439 For the purpose of this analysis, modules are viewed as interacting in two ways:

a) to provide services to one another, and

b) to cooperate in support of security functions.

1440 The low-level design should include specific information on these interrelationships. For example, if a module performs calculations that depend on the results of calculations in other modules, those other modules should be listed. Further, if a module provides a service intended for other modules to use in supporting security functions, the service should be described. It is possible that the description of the purpose of a module, as analysed in the preceding work unit, is sufficient to provide this information.

ADV_LLD.1.6C

4:ADV_LLD.1-6 The evaluator ***shall examine*** the low-level design to determine that it describes how each of the TSP-enforcing functions is provided.

1441 The TSP-enforcing functions are those functions of the TSF that directly or indirectly enforce the TSP.

1442 It is this description in the low-level design that is key to the assessment as to whether the low-level design is sufficiently refined to permit an implementation to be created. The evaluator should analyse the description from the point of view of an implementor. If the evaluator, using the implementor's viewpoint, is unclear on any aspect of how the module could be implemented, the description is incomplete. Note that there is no requirement that a module be implemented as a separate unit (be it a program, a subprogram, or a hardware component); but the low-level design may be sufficiently detailed to permit such an implementation.

ADV_LLD.1.7C

4:ADV_LLD.1-7 The evaluator ***shall check*** that the low-level design identifies the interfaces to the TSF modules.

1443 The low-level design should include, for each module, the name of each of its entry points.

ADV_LLD.1.8C

4:ADV_LLD.1-8 The evaluator ***shall check*** that the low-level design identifies which of the interfaces to the modules of the TSF are externally visible.

1444 As discussed under work unit 4:ADV_FSP.2-3, external interfaces (i.e. those visible to the user) may directly or indirectly access the TSF. Any external interface that accesses the TSF either directly or indirectly is included in the identification for this work unit. External interfaces that do not access the TSF need not be included.

ADV_LLD.1.9C

4:ADV_LLD.1-9 The evaluator ***shall examine*** the low-level design to determine that it describes the interfaces to each module in terms of their purpose and method of use, and provides details of effects, exceptions and error messages, as appropriate.

1445 The module interface descriptions may be provided in general terms for some interfaces, and in more detail for others. In determining the necessary level of detail of effects, exceptions and error messages, the evaluator should consider the purposes of this analysis and the uses made of the interface by the TOE. For example, the evaluator needs to understand the general nature of the interactions between modules to establish confidence that the TOE design is sound, and may be able to obtain this understanding with only a general description of some of the interfaces between modules. In particular, internal entry points that are not called by any other module would not normally require detailed descriptions.

1446 This work unit may be performed in conjunction with the evaluator's independent vulnerability analysis, which is part of the AVA_VLA sub-activity.

1447 Detailed descriptions would include details of any input and output parameters, of the effects of the interface, and of any exceptions or error messages it produces. In the case of external interfaces, the required description is probably included in the functional specification and can be referenced in the low-level design without replication.

ADV_LLD.1.10C

4:ADV_LLD.1-10 The evaluator ***shall check*** that the low-level design describes the separation of the TOE into TSP-enforcing and other modules.

1448 The TSF comprises all the parts of the TOE that have to be relied upon for enforcement of the TSP. Because the TSF includes both functions that directly enforce the TSP, and also those functions that, while not directly enforcing the TSP, contribute to the enforcement of the TSP in a more indirect manner, all TSP-enforcing modules are contained in the TSF. Modules that cannot affect TSP enforcement are not part of the TSF.

8.6.5.3.2 Action ADV_LLD.1.2E

4:ADV_LLD.1-11 The evaluator ***shall examine*** the low-level design to determine that it is an accurate instantiation of the TOE security functional requirements.

1449 The evaluator validates the module interface specifications by ensuring that:

a) the interface specifications are consistent with the description of the purpose of the module;

b) the interface specifications are consistent with their use by other modules;

c) the interrelationships between modules that are needed in order that each TSP-enforcing function is correctly supported are correctly stated.

4:ADV_LLD.1-12 The evaluator ***shall examine*** the low-level design to determine that it is a complete instantiation of the TOE security functional requirements.

1450 The evaluator ensures that all ST functional requirements are mapped onto applicable sections of the low-level design. This determination should be made in conjunction with the ADV_RCR.1 sub-activity.

1451 The evaluator analyses the low-level design to determine that each TOE security function is completely described by the module specifications, and that there are no modules on which a TOE security function relies for which there is no specification in the low-level design.

8.6.6 Evaluation of representation correspondence (ADV_RCR.1)

8.6.6.1 Objectives

1452 The objective of this sub-activity is to determine whether the developer has correctly and completely implemented the requirements of the ST, functional specification, high-level design and low-level design in the implementation representation.

8.6.6.2 Input

1453 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the low-level design;

e) the subset of the implementation representation;

f) the correspondence analysis between the TOE summary specification and the functional specification;

g) the correspondence analysis between the functional specification and the high-level design;

h) the correspondence analysis between the high-level design and the low-level design;

i) the correspondence analysis between the low-level design and the subset of the implementation representation.

8.6.6.3 Evaluator actions

1454 This sub-activity comprises one CC Part 3 evaluator action element:

a) ADV_RCR.1.1E.

8.6.6.3.1 Action ADV_RCR.1.1E

4:ADV_RCR.1-1 The evaluator ***shall examine*** the correspondence analysis between the TOE summary specification and the functional specification to determine that the functional specification is a correct and complete representation of the TOE security functions.

1455 The evaluator's goal in this work unit is to determine that all security functions identified in the TOE summary specification are represented in the functional specification and that they are represented accurately.

1456 The evaluator reviews the correspondence between the TOE security functions of the TOE summary specification and the functional specification. The evaluator looks for consistency and accuracy in the correspondence. Where the correspondence analysis indicates a relationship between a security function of the TOE summary specification and an interface description in the functional specification, the evaluator verifies that the security functionality of both are the same. If the security functions of the TOE summary specification are correctly and completely present in the corresponding interface, this work unit will be satisfied.

1457 This work unit may be done in conjunction with work units 4:ADV_FSP.2-8 and 4:ADV_FSP.2-9.

4:ADV_RCR.1-2 The evaluator ***shall examine*** the correspondence analysis between the functional specification and the high-level design to determine that the high-level design is a correct and complete representation of the functional specification.

1458 The evaluator uses the correspondence analysis, the functional specification, and the high-level design to ensure that it is possible to map each security function identified in the functional specification onto a TSF subsystem described in the high-level design. For each security function, the correspondence indicates which TSF subsystems are involved in the support of the function. The evaluator verifies that the high-level design includes a description of a correct realisation of each security function.

4:ADV_RCR.1-3 The evaluator ***shall examine*** the correspondence analysis between the high-level design and the low-level design to determine that the low-level design is a correct and complete representation of the high-level design.

1459 The evaluator uses the correspondence analysis, the high-level design, and the low-level design to ensure that it is possible to map each TSF module identified in the low-level design onto a TSF subsystem described in the high-level design. For each TOE security function, the correspondence indicates which TSF modules are involved in the support of the function. The evaluator verifies that the low-level design includes a description of a correct realisation of each security function.

4:ADV_RCR.1-4 The evaluator ***shall examine*** the correspondence analysis between the low-level design and the subset of the implementation representation to determine that the subset is a correct and complete representation of those portions of the low-level design that are refined in the implementation representation.

1460 Since the evaluator examines only a subset of the implementation representation, this work unit is performed by assessing the correspondence analysis of the subset of the implementation representation to the relevant parts of the low-level design rather than attempting to trace each TOE security function into the implementation representation. The subset may provide no coverage for some functions.

8.6.7 Evaluation of security policy modeling (ADV_SPM.1)

8.6.7.1 Objectives

1461 The objectives of this sub-activity are to determine whether the security policy model clearly and consistently describes the rules and characteristics of the security policies and whether this description corresponds with the description of security functions in the functional specification.

8.6.7.2 Input

1462 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE security policy model;

d) the user guidance;

e) the administrator guidance.

8.6.7.3 Evaluator Actions

1463 This sub-activity comprises one CC Part 3 evaluator action element:

a) ADV_SPM.1.1E.

8.6.7.3.1 Action ADV_SPM.1.1E

ADV_SPM.1.1C

4:ADV_SPM.1-1 The evaluator ***shall examine*** the security policy model to determine that it contains all necessary informal explanatory text.

1464 If the entire security policy model is informal, this work unit is not applicable and is therefore considered to be satisfied.

1465 Supporting narrative descriptions are necessary for those portions of the security policy model that are difficult to understand only from the semiformal or formal description (for example, to make clear the meaning of any formal notation).

ADV_SPM.1.2C

4:ADV_SPM.1-2 The evaluator ***shall check*** the security policy model to determine that all security policies that are explicitly included in the ST are modeled.

1466 The security policy is expressed by the collection of the functional security requirements in the ST. Therefore, to determine the nature of the security policy (and hence what policies must be modeled), the evaluator analyses the ST functional requirements for those policies explicitly called for (by FDP_ACC and FDP_IFC, if included in the ST).

1467 Depending upon the TOE, formal/semiformal modeling might not even be possible for access control. (For example, the access control policy for a firewall connected to the internet cannot be formally modeled in a useful manner because the state of the internet cannot be completely defined.). For any security policy where formal or semiformal models are not possible, the policy must be provided in an informal form.

1468 If the ST contains no explicit policies (because neither FDP_ACC nor FDP_IFC are included in the ST), this work unit is not applicable and is therefore considered to be satisfied.

4:ADV_SPM.1-3 The evaluator ***shall examine*** the security policy model to determine that all security policies represented by the security functional requirements claimed in the ST are modeled.

1469 In addition to the explicitly-listed policies (see work unit 4:ADV_SPM.1-2), the evaluator analyses the ST functional requirements for those policies implied by the other functional security requirement classes. For example, inclusion of FDP requirements (other than FDP_ACC and FDP_IFC) would need a description of the Data Protection policy being enforced; inclusion of any FIA requirements would necessitate that a description of the Identification and Authentication policies be present in the security policy model; inclusion of FAU requirements need a description of the Audit policies; etc. While the other functional requirement families are not typically associated with what are commonly referred to as *security policies*, they nevertheless do enforce security policies (e.g. non-repudiation, reference mediation, privacy, etc.) that must be included in the security policy model.

1470 In cases where the security policy model presentation is informal, all security policies can be modeled (i.e. described), and so must be included. For any security policy where formal or semiformal security policy models are not possible, the policy must be provided in an informal form.

1471 If the ST contains no such implicit policies, this work unit is not applicable and is therefore considered to be satisfied.

4:ADV_SPM.1-4 The evaluator ***shall examine*** the rules and characteristics of the security policy model to determine that the modeled security behaviour of the TOE is clearly articulated.

1472 The rules and characteristics describe the security posture of the TOE. It is likely that such a description would be contained within an evaluated and certified ST. In order to be considered a clear articulation, such a description should define the notion of security for the TOE, identify the security attributes of the entities controlled by the TOE and identify the TOE actions which change those attributes.

For example, if a policy attempts to address data integrity concerns, the security policy model would:

a) define the notion of integrity for that TOE;

b) identify the types of data for which the TOE would maintain integrity;

c) identify the entities that could modify that data;

d) identify the rules that potential modifiers must follow to modify data.

ADV_SPM.1.3C

4:ADV_SPM.1-5 The evaluator ***shall examine*** the security policy model rationale to determine that the behaviour modeled is consistent with respect to policies described by the security policies (as articulated by the functional requirements in the ST).

1473 In determining consistency, the evaluator verifies that the rationale shows that each rule or characteristic description in the security policy model accurately reflects the intent of the security policies. For example, if a policy stated that access control was necessary to the granularity of a single individual, then a security policy model describing the security behaviour of a TOE in the context of controlling groups of users would not be consistent. Likewise, if the policy stated that access control for groups of users was necessary, then a security policy model describing the security behaviour of a TOE in the context of controlling individual users would also not be consistent.

Interp Note : The following three paragraphs are added as a result of Interpretation 069.

Assurance is to be gained from an explicit and general statement of the policies underlying the TOE security functional requirements. The assurance gained is two-fold: collecting the description of each security policy into a concise whole aids in understanding the details of the policies being enforced. Additionally, such a collected description makes it much easier to see any gaps or inconsistencies (which must be sought as part of the ADV_SPM.*.3C element), and provides a clear characterisation of secure states (sought as part of the ADV_SPM.*.2C element).

The requirement for an Informal Security Policy Model (ISPM) is met by a clear statement of the security policy. The need for a separate ISPM is not absolute, since for very straightforward policies, or those very clearly expressed in the ST, there may be no need for a separate ISPM. In such cases, different sections of the ST (e.g. security requirements, TOE summary specification) may combine together to provide a sufficient level of detail for the security policy. However, this is often not the case. For example, audit requirements may be spread throughout the statement of TOE security functional requirements, which may not provide a clear model of the overall policy. Unless another section of the ST (perhaps the TOE summary specification) pulls together the audit requirements into a cohesive whole, then having a separate ISPM would be necessary in order to allow for the detection of inconsistencies within the ST requirements that may otherwise pass undetected.

Where a developer claims that the ISPM requirements for some or all of the security policies are met by the ST, the evaluator needs to determine that this is the case by applying the requirements of the ADV_SPM.1 component: determining that the policy is clearly expressed, and that the model is consistent with the remainder of the ST. As part of the ISPM rationale, it is likely that, in cases where the developer claims that the ISPM is met entirely by the ST, that the rationale will reference the demonstrations of suitability and correspondence between portions of the ST. When evaluating this work-unit, the evaluator may draw upon the results of the ST evaluation in this area.

1474 For guidance on consistency analysis see Annex B.3.

4:ADV_SPM.1-6 The evaluator ***shall examine*** the security policy model rationale to determine that the behaviour modeled is complete with respect to the policies described by the security policies (i.e. as articulated by the functional requirements in the ST).

1475 In determining completeness of this rationale, the evaluator considers the rules and characteristics of the security policy model and maps those rules and characteristics to explicit policy statements (i.e. functional requirements). The rationale should show that all policies that are required to be modeled have an associated rule or characteristic description in the security policy model.

Interp Note : The following paragraph is added as a result of Interpretation 069.

Where a developer claims that the ISPM requirements for some or all of the security policies are met by the ST, the evaluator needs to determine that this is the case by applying the requirements of the ADV_SPM.1 component: determining that the policy is clearly expressed, and that the model is complete with respect to the remainder of the ST. When evaluating this work-unit, the evaluator may draw upon the results of the evaluation of the completeness of the various portions of the ST.

ADV_SPM.1.4C

4:ADV_SPM.1-7 The evaluator ***shall examine*** the functional specification correspondence demonstration of the security policy model to determine that it identifies all security functions described in the functional specification that implement a portion of the policy.

1476 In determining completeness, the evaluator reviews the functional specification, identifies which functions directly support the security policy model and verifies that these functions are present in the functional specification correspondence demonstration of the security policy model.

4:ADV_SPM.1-8 The evaluator ***shall examine*** the functional specification correspondence demonstration of the security policy model to determine that the descriptions of the functions identified as implementing the security policy model are consistent with the descriptions in the functional specification.

1477 To demonstrate consistency, the evaluator verifies that the functional specification correspondence shows that the functional description in the functional specification of the functions identified as implementing the policy described in the security policy model identify the same attributes and characteristics of the security policy model and enforce the same rules as the security policy model.

1478 In cases where a security policy is enforced differently for untrusted users and administrators, the policies for each are described consistently with the respective behaviour descriptions in the user and administrator guidance. For example, the “identification and authentication” policy enforced upon remote untrusted users might be more stringent than that enforced upon administrators whose only point of access is within a physically-protected area; the differences in authentication should correspond to the differences in the descriptions of authentication within the user and administrator guidance.

1479 For guidance on consistency analysis see Annex B.3.

8.7 Guidance documents activity

1480 The purpose of the guidance document activity is to judge the adequacy of the documentation describing how to use the operational TOE. Such documentation includes both that aimed at trusted administrators and non-administrator users whose incorrect actions could adversely affect the security of the TOE, as well as that aimed at untrusted users whose incorrect actions could adversely affect the security of their own data.

1481 The guidance documents activity at EAL4 contains sub-activities related to the following components:

a) AGD_ADM.1;

b) AGD_USR.1.

8.7.1 Application notes

1482 The guidance documents activity applies to those functions and interfaces which are related to the security of the TOE. The secure configuration of the TOE is described in the ST.

8.7.2 Evaluation of administrator guidance (AGD_ADM.1)

8.7.2.1 Objectives

1483 The objective of this sub-activity is to determine whether the administrator guidance describes how to administer the TOE in a secure manner.

8.7.2.2 Application notes

1484 The term *administrator* is used to indicate a human user who is trusted to perform security critical operations within the TOE, such as setting TOE configuration parameters. The operations may affect the enforcement of the TSP, and the administrator therefore possesses specific privileges necessary to perform those operations. The role of the administrator(s) has to be clearly distinguished from the role of non-administrative users of the TOE.

1485 There may be different administrator roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF such as auditor, administrator, or daily-management. Each role can encompass an extensive set of capabilities, or can be a single one. The capabilities of these roles and their associated privileges are described in the FMT class. Different administrator roles and groups should be taken into consideration by the administrator guidance.

8.7.2.3 Input

1486 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures;

g) the life-cycle definition.

8.7.2.4 Evaluator actions

1487 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_ADM.1.1E.

8.7.2.4.1 Action AGD_ADM.1.1E

AGD_ADM.1.1C

4:AGD_ADM.1-1 The evaluator ***shall examine*** the administrator guidance to determine that it describes the administrative security functions and interfaces available to the administrator of the TOE.

1488 The administrator guidance should contain an overview of the security functionality that is visible at the administrator interfaces.

1489 The administrator guidance should identify and describe the purpose, behaviour, and interrelationships of the administrator security interfaces and functions.

1490 For each administrator security interface and function, the administrator guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system calls, menu selection, command button);

b) describe the parameters to be set by the administrator, their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_ADM.1.2C

4:AGD_ADM.1-2 The evaluator ***shall examine*** the administrator guidance to determine that it describes how to administer the TOE in a secure manner.

1491 The administrator guidance describes how to operate the TOE according to the TSP in an IT environment that is consistent with the one described in the ST.

AGD_ADM.1.3C

4:AGD_ADM.1-3 The evaluator ***shall examine*** the administrator guidance to determine that it contains warnings about functions and privileges that should be controlled in a secure processing environment.

1492 The configuration of the TOE may allow users to have dissimilar privileges to make use of the different functions of the TOE. This means that some users may be authorised to perform certain functions while other users may not be so authorised. These functions and privileges should be described by the administrator guidance.

1493 The administrator guidance identifies the functions and privileges that must be controlled, the types of controls required for them, and the reasons for such controls. Warnings address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_ADM.1.4C

4:AGD_ADM.1-4 The evaluator ***shall examine*** the administrator guidance to determine that it describes all assumptions regarding user behaviour that are relevant to the secure operation of the TOE.

1494 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the information that is of concern to the secure operation of the TOE need be included in the administrator guidance.

1495 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

AGD_ADM.1.5C

4:AGD_ADM.1-5 The evaluator ***shall examine*** the administrator guidance to determine that it describes all security parameters under the control of the administrator indicating secure values as appropriate.

1496 For each security parameter, the administrator guidance should describe the purpose of the parameter, the valid and default values of the parameter, and secure and insecure use settings of such parameters, both individually or in combination.

AGD_ADM.1.6C

4:AGD_ADM.1-6 The evaluator ***shall examine*** the administrator guidance to determine that it describes each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

1497 All types of security-relevant events are detailed, such that an administrator knows what events may occur and what action (if any) the administrator may have to take in order to maintain security. Security-relevant events that may occur during operation of the TOE (e.g. audit trail overflow, system crash, updates to user records, such as when a user account is removed when the user leaves the organisation) are adequately defined to allow administrator intervention to maintain secure operation.

AGD_ADM.1.7C

4:AGD_ADM.1-7 The evaluator ***shall examine*** the administrator guidance to determine that it is consistent with all other documents supplied for evaluation.

1498 The ST in particular may contain detailed information on any warnings to the TOE administrators with regard to the TOE security environment and the security objectives.

1499 For guidance on consistency analysis see Annex B.3.

AGD_ADM.1.8C

4:AGD_ADM.1-8 The evaluator ***shall examine*** the administrator guidance to determine that it describes all IT security requirements for the IT environment of the TOE that are relevant to the administrator.

1500 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

1501 This work unit relates to IT security requirements only and not to any organisational security policies.

1502 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare them with the administrator guidance to ensure that all security requirements of the ST that are relevant to the administrator are described appropriately in the administrator guidance.

8.7.3 Evaluation of user guidance (AGD_USR.1)

8.7.3.1 Objectives

1503 The objectives of this sub-activity are to determine whether the user guidance describes the security functions and interfaces provided by the TSF and whether this guidance provides instructions and guidelines for the secure use of the TOE.

8.7.3.2 Application notes

1504 There may be different user roles or groups defined in the ST that are recognised by the TOE and that can interact with the TSF. The capabilities of these roles and their associated privileges are described in the FMT class. Different user roles and groups should be taken into consideration by the user guidance.

8.7.3.3 Input

1505 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the user guidance;

e) the administrator guidance;

f) the secure installation, generation, and start-up procedures.

8.7.3.4 Evaluator actions

1506 This sub-activity comprises one CC Part 3 evaluator action element:

a) AGD_USR.1.1E.

8.7.3.4.1 Action AGD_USR.1.1E

AGD_USR.1.1C

4:AGD_USR.1-1 The evaluator ***shall examine*** the user guidance to determine that it describes the security functions and interfaces available to the non-administrative users of the TOE.

1507 The user guidance should contain an overview of the security functionality that is visible at the user interfaces.

1508 The user guidance should identify and describe the purpose of the security interfaces and functions.

AGD_USR.1.2C

4:AGD_USR.1-2 The evaluator ***shall examine*** the user guidance to determine that it describes the use of user-accessible security functions provided by the TOE.

1509 The user guidance should identify and describe the behaviour and interrelationship of the security interfaces and functions available to the user.

1510 If the user is allowed to invoke a TOE security function, the user guidance provides a description of the interfaces available to the user for that function.

1511 For each interface and function, the user guidance should:

a) describe the method(s) by which the interface is invoked (e.g. command-line, programming-language system call, menu selection, command button);

b) describe the parameters to be set by the user and their valid and default values;

c) describe the immediate TSF response, message, or code returned.

AGD_USR.1.3C

4:AGD_USR.1-3 The evaluator ***shall examine*** the user guidance to determine that it contains warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

1512 The configuration of the TOE may allow users to have dissimilar privileges in making use of the different functions of the TOE. This means that some users are authorised to perform certain functions, while other users may not be so authorised. These user-accessible functions and privileges are described by the user guidance.

1513 The user guidance should identify the functions and privileges that can be used, the types of commands required for them, and the reasons for such commands. The user guidance should contain warnings regarding the use of the functions and privileges that must be controlled. Warnings should address expected effects, possible side effects, and possible interactions with other functions and privileges.

AGD_USR.1.4C

4:AGD_USR.1-4 The evaluator ***shall examine*** the user guidance to determine that it presents all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

1514 Assumptions about the user behaviour may be described in more detail in the statement of the TOE security environment of the ST. However, only the

information that is of concern to the secure operation of the TOE need be included in the user guidance.

1515 The user guidance should provide advice regarding effective use of the security functions (e.g. reviewing password composition practices, suggested frequency of user file backups, discussion on the effects of changing user access privileges).

1516 An example of a user's responsibility necessary for secure operation is that users will keep their passwords secret.

1517 The user guidance should indicate whether the user can invoke a function or whether the user requires the assistance of an administrator.

AGD_USR.1.5C

4:AGD_USR.1-5 The evaluator ***shall examine*** the user guidance to determine that it is consistent with all other documentation supplied for evaluation.

1518 The evaluator ensures that the user guidance and all other documents supplied for evaluation do not contradict each other. This is especially true if the ST contains detailed information on any warnings to the TOE users with regard to the TOE security environment and the security objectives.

1519 For guidance on consistency analysis see Annex B.3.

AGD_USR.1.6C

4:AGD_USR.1-6 The evaluator ***shall examine*** the user guidance to determine that it describes all security requirements for the IT environment of the TOE that are relevant to the user.

1520 If the ST does not contain IT security requirements for the IT environment, this work unit is not applicable, and is therefore considered to be satisfied.

1521 This work unit relates to IT security requirements only and not to any organisational security policies.

1522 The evaluator should analyse the security requirements for the IT environment of the TOE (optional statement in the ST) and compare that with the user guidance to ensure that all security requirements of the ST, that are relevant to the user, are described appropriately in the user guidance.

8.8 Life-cycle support activity

1523 The purpose of the life-cycle support activity is to determine the adequacy of the procedures the developer uses during the development and maintenance of the TOE. These procedures include the security measures used throughout TOE development, the life-cycle model used by the developer, and the tools used by the developer throughout the life-cycle of the TOE.

1524 Developer security procedures are intended to protect the TOE and its associated design information from interference or disclosure. Interference in the development process may allow the deliberate introduction of vulnerabilities. Disclosure of design information may allow vulnerabilities to be more easily exploited. The adequacy of the procedures will depend on the nature of the TOE and the development process.

1525 Poorly controlled development and maintenance of the TOE can result in vulnerabilities in the implementation. Conformance to a defined life-cycle model can help to improve controls in this area.

1526 The use of well-defined development tools helps to ensure that vulnerabilities are not inadvertently introduced during refinement.

1527 The life-cycle support activity at EAL4 contains sub-activities related to the following components:

a) ALC_DVS.1;

b) ALC_LCD.1;

c) ALC_TAT.1.

8.8.1 Evaluation of development security (ALC_DVS.1)

8.8.1.1 Objectives

1528 The objective of this sub-activity is to determine whether the developer's security controls on the development environment are adequate to provide the confidentiality and integrity of the TOE design and implementation that is necessary to ensure that secure operation of the TOE is not compromised.

8.8.1.2 Input

1529 The evaluation evidence for this sub-activity is:

a) the ST;

b) the development security documentation.

1530 In addition, the evaluator may need to examine other deliverables to determine that the security controls are well-defined and followed. Specifically, the evaluator may need to examine the developer's configuration management documentation (the input for the ACM_CAP.4 and ACM_SCP.2 sub-activities). Evidence that the procedures are being applied is also required.

8.8.1.3 Evaluator actions

1531 This sub-activity comprises two CC Part 3 evaluator action elements:

a) ALC_DVS.1.1E;

b) ALC_DVS.1.2E.

8.8.1.3.1 Action ALC_DVS.1.1E

ALC_DVS.1.1C

4:ALC_DVS.1-1 The evaluator ***shall examine*** the development security documentation to determine that it details all security measures used in the development environment that are necessary to protect the confidentiality and integrity of the TOE design and implementation.

1532 The evaluator determines what is necessary by first referring to the ST for any information that may assist in the determination of necessary protection, especially the sections on threats, organisational security policies and assumptions, although there may be no information provided explicitly. The statement of security objectives for the environment may also be useful in this respect.

1533 If no explicit information is available from the ST the evaluator will need to make a determination of the necessary measures, based upon a consideration of the intended environment for the TOE. In cases where the developer's measures are considered less than what is necessary, a clear justification should be provided for the assessment, based on a potential exploitable vulnerability.

1534 The following types of security measures are considered by the evaluator when examining the documentation:

a) *physical*, for example physical access controls used to prevent unauthorised access to the TOE development environment (during normal working hours and at other times);

b) *procedural*, for example covering:

- granting of access to the development environment or to specific parts of the environment such as development machines
- revocation of access rights when a person leaves the development team
- transfer of protected material out of the development environment

- admitting and escorting visitors to the development environment
- roles and responsibilities in ensuring the continued application of security measures, and the detection of security breaches.

c) *personnel*, for example any controls or checks made to establish the trustworthiness of new development staff;

d) *other security measures*, for example the logical protections on any development machines.

1535 The development security documentation should identify the locations at which development occurs, and describe the aspects of development performed, along with the security measures applied at each location. For example, development could occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites. Development includes such tasks as creating multiple copies of the TOE, where applicable. This work-unit should not overlap with those for ADO_DEL, but the evaluator should ensure that all aspects are covered by one sub-activity or the other.

1536 In addition, the development security documentation may describe different security measures that can be applied to different aspects of development in terms of their performance and the required inputs and outputs. For example, different procedures may be applicable to the development of different portions of the TOE, or to different stages of the development process.

4:ALC_DVS.1-2 The evaluator ***shall examine*** the development confidentiality and integrity policies in order to determine the sufficiency of the security measures employed.

1537 These include the policies governing:

a) what information relating to the TOE development needs to be kept confidential, and which members of the development staff are allowed to access such material;

b) what material must be protected from unauthorised modification in order to preserve the integrity of the TOE, and which members of the development staff are allowed to modify such material.

1538 The evaluator should determine that these policies are described in the development security documentation, that the security measures employed are consistent with the policies, and that they are complete.

1539 It should be noted that configuration management procedures will help protect the integrity of the TOE and the evaluator should avoid overlap with the work-units conducted for the ACM_CAP sub-activity. For example, the CM documentation may describe the security procedures necessary for controlling the roles or individuals who should have access to the development environment and who may modify the TOE.

1540 Whereas the ACM_CAP requirements are fixed, those for ALC_DVS, mandating only necessary measures, are dependent on the nature of the TOE, and on information that may be provided in the Security Environment section of the ST. For example, the ST may identify an organisational security policy that requires the TOE to be developed by staff who have security clearance. The evaluators would then determine that such a policy had been applied under this sub-activity.

ALC_DVS.1.2C

4:ALC_DVS.1-3 The evaluator ***shall check*** the development security documentation to determine that documentary evidence that would be produced as a result of application of the procedures has been generated.

1541 Where documentary evidence is produced the evaluator inspects it to ensure compliance with procedures. Examples of the evidence produced may include entry logs and audit trails. The evaluator may choose to sample the evidence.

1542 For guidance on sampling see Annex B.2.

8.8.1.3.2 Action ALC_DVS.1.2E

4:ALC_DVS.1-4 The evaluator ***shall examine*** the development security documentation and associated evidence to determine that the security measures are being applied.

1543 This work unit requires the evaluator to determine that the security measures described in the development security documentation are being followed, such that the integrity of the TOE and the confidentiality of associated documentation is being adequately protected. For example, this could be determined by examination of the documentary evidence provided. Documentary evidence should be supplemented by visiting the development environment. A visit to the development environment will allow the evaluator to:

a) observe the application of security measures (e.g. physical measures);

b) examine documentary evidence of application of procedures;

c) interview development staff to check awareness of the development security policies and procedures, and their responsibilities.

1544 A development site visit is a useful means of gaining confidence in the measures being used. Any decision not to make such a visit should be determined in consultation with the overseer.

1545 For guidance on site visits see Annex B.5.

8.8.2 Evaluation of life-cycle definition (ALC_LCD.1)

8.8.2.1 Objectives

1546 The objective of this sub-activity is to determine whether the developer has used a documented model of the TOE life-cycle.

8.8.2.2 Input

1547 The evaluation evidence for this sub-activity is:

a) the ST;

b) the life-cycle definition documentation.

8.8.2.3 Evaluator actions

1548 This sub-activity comprises one CC Part 3 evaluator action element:

a) ALC_LCD.1.1E.

8.8.2.3.1 Action ALC_LCD.1.1E

ALC_LCD.1.1C

4:ALC_LCD.1-1 The evaluator ***shall examine*** the documented description of the life-cycle model used to determine that it covers the development and maintenance process.

1549 A life-cycle model encompasses the procedures, tools and techniques used to develop and maintain the TOE. The description of the life-cycle model should include information on the procedures, tools and techniques used by the developer (e.g. for design, coding, testing, bug-fixing). It should describe overall management structure governing the application of the procedures (e.g. an identification and description of the individual responsibilities for each of the procedures required by the development and maintenance process covered by the life-cycle model). ALC_LCD.1 does not require the model used to conform to any standard life-cycle model.

ALC_LCD.1.2C

4:ALC_LCD.1-2 The evaluator ***shall examine*** the life-cycle model to determine that use of the procedures, tools and techniques described by the life-cycle model will make the necessary positive contribution to the development and maintenance of the TOE.

1550 The information provided in the life-cycle model gives the evaluator assurance that the development and maintenance procedures adopted would minimise the likelihood of security flaws. For example, if the life-cycle model described the review process, but did not make provision for recording changes to components, then the evaluator may be less confident that errors will not be introduced into the TOE. The evaluator may gain further assurance by comparing the description of the model against an understanding of the development process gleaned from performing other evaluator actions relating to the TOE development (e.g. those actions covered under the ACM activity). Identified deficiencies in the life-cycle model will be of concern if they might reasonably be expected to give rise to the introduction of flaws into the TOE, either accidentally or deliberately.

1551 The CC does not mandate any particular development approach, and each should be judged on merit. For example, spiral, rapid-prototyping and waterfall approaches to design can all be used to produce a quality TOE if applied in a controlled environment.

8.8.3 Evaluation of tools and techniques (ALC_TAT.1)

8.8.3.1 Objectives

1552 The objective of this sub-activity is to determine whether the developer has used well-defined development tools (e.g. programming languages or computer-aided design (CAD) systems) that yield consistent and predictable results.

8.8.3.2 Input

1553 The evaluation evidence for this sub-activity is:

a) the development tool documentation;

b) the subset of the implementation representation.

8.8.3.3 Application note

1554 This work may be performed in parallel with the ADV_IMP.1 sub-activity, specifically with regard to determining the use of features in the tools that will affect the object code (e.g. compilation options).

8.8.3.4 Evaluator actions

1555 This sub-activity comprises one CC Part 3 evaluator action element:

a) ALC_TAT.1.1E.

8.8.3.4.1 Action ALC_TAT.1.1E

ALC_TAT.1.1C

4:ALC_TAT.1-1 The evaluator ***shall examine*** the development tool documentation provided to determine that all development tools are well-defined.

1556 For example, a well-defined language, compiler or CAD system may be considered to be one that conforms to a recognised standard, such as the ISO standards. A well-defined language is one that has a clear and complete description of its syntax, and a detailed description of the semantics of each construct.

ALC_TAT.1.2C

4:ALC_TAT.1-2 The evaluator ***shall examine*** the documentation of development tools to determine that it unambiguously defines the meaning of all statements used in the implementation.

1557 The development tool documentation (e.g. programming language specifications and user manuals) should cover all statements used in the implementation representation of the TOE, and for each such statement provide a clear and unambiguous definition of the purpose and effect of that statement. This work may be performed in parallel with the evaluator's examination of the implementation representation performed during the ADV_IMP.1 sub-activity. The key test the evaluator should apply is whether or not the documentation is sufficiently clear for the evaluator to be able to understand the implementation representation. The documentation should not assume (for example) that the reader is an expert in the programming language used.

1558 Reference to the use of a documented standard is an acceptable approach to meet this requirement, provided that the standard is available to the evaluator. Any differences from the standard should be documented.

1559 The critical test is whether the evaluator can understand the TOE source code when performing source code analysis covered in the ADV_IMP sub-activity. However, the following checklist can additionally be used in searching for problem areas:

a) In the language definition, phrases such as "the effect of this construct is undefined", and terms such as "implementation dependent" or "erroneous" may indicate ill-defined areas;

b) Aliasing (allowing the same piece of memory to be referenced in different ways) is a common source of ambiguity problems;

c) Exception handling (e.g. what happens after memory exhaustion or stack overflow) is often poorly defined.

1560 Most languages in common use, however well designed, will have some problematic constructs. If the implementation language is mostly well defined, but some problematic constructs exist, then an inconclusive verdict should be assigned, pending examination of the source code.

1561 The evaluator should verify, during the examination of source code, that any use of the problematic constructs does not introduce vulnerabilities. The evaluator should also ensure that constructs precluded by the documented standard are not used.

ALC_TAT.1.3C

4:ALC_TAT.1-3 The evaluator ***shall examine*** the development tool documentation to determine that it unambiguously defines the meaning of all implementation-dependent options.

1562 The documentation of software development tools should include definitions of implementation-dependent options that may affect the meaning of the executable code, and those that are different from the standard language as documented. Where source code is provided to the evaluator, information should also be provided on compilation and linking options used.

1563 The documentation for hardware design and development tools should describe the use of all options that affect the output from the tools (e.g. detailed hardware specifications, or actual hardware).

8.9 Tests activity

1564 The purpose of this activity is to determine whether the TOE behaves as specified in the design documentation and in accordance with the TOE security functional requirements specified in the ST. This is accomplished by determining that the developer has tested the TSF against its functional specification and high-level design, gaining confidence in those test results by performing a sample of the developer's tests, and by independently testing a subset of the TSF.

1565 The tests activity at EAL4 contains sub-activities related to the following components:

a) ATE_COV.2;

b) ATE_DPT.1;

c) ATE_FUN.1;

d) ATE_IND.2.

8.9.1 Application notes

1566 The size and composition of the evaluator's test subset depends upon several factors discussed in the independent testing (ATE_IND.2) sub-activity. One such factor affecting the composition of the subset is *known public domain weaknesses,* information about which the evaluator needs access (e.g. from a scheme).

1567 The CC has separated coverage and depth from functional tests to increase the flexibility when applying the components of the families. However, the requirements of the families are intended to be applied together to confirm that the TSF operates according to its specification. This tight coupling of families has led to some duplication of evaluator work effort across sub-activities. These application notes are used to minimize duplication of text between sub-activities of the same activity and EAL.

8.9.1.1 Understanding the expected behaviour of the TOE

1568 Before the adequacy of test documentation can be accurately evaluated, or before new tests can be created, the evaluator has to understand the desired expected behaviour of a security function in the context of the requirements it is to satisfy.

1569 The evaluator may choose to focus on one security function of the TSF at a time. For each security function, the evaluator examines the ST requirement and the relevant parts of the functional specification, high-level design and guidance documentation to gain an understanding of the way the TOE is expected to behave.

1570 With an understanding of the expected behaviour, the evaluator examines the test plan to gain an understanding of the testing approach. In most cases, the testing approach will entail a security function being stimulated at either the external or internal interfaces and its responses are observed. However, there may be cases where a security function cannot be adequately tested at an interface (as may be the case, for instance, for residual information protection functionality); in such cases, other means will need to be employed.

8.9.1.2 Testing vs. alternate approaches to verify the expected behaviour of a security function

1571 In cases where it is impractical or inadequate to test at an interface, the test plan should identify the alternate approach to verify expected behaviour. It is the evaluator's responsibility to determine the suitability of the alternate approach. However, the following should be considered when assessing the suitability of alternate approaches:

a) an analysis of the implementation representation to determine that the required behaviour should be exhibited by the TOE is an acceptable alternate approach. This could mean a code inspection for a software TOE or perhaps a chip mask inspection for a hardware TOE.

b) it is acceptable to use evidence of developer integration or module testing, even if the EAL is not commensurate with evaluation exposure to the low-level design or implementation. If evidence of developer integration or module testing is used in verifying the expected behaviour of a security function, care should be given to confirm that the testing evidence reflects the current implementation of the TOE. If the subsystem or modules have been changed since testing occurred, evidence that the changes were tracked and addressed by analysis or further testing will usually be required.

1572 It should be emphasized that supplementing the testing effort with alternate approaches should only be undertaken when both the developer and evaluator determine that there exists no other practical means to test the expected behaviour of a security function. This alternative is made available to the developer to minimize the cost (time and/or money) of testing under the circumstances described above; it is not designed to give the evaluator more latitude to demand unwarranted additional information about the TOE, nor to replace testing in general.

8.9.1.3 Verifying the adequacy of tests

1573 Test prerequisites are necessary to establish the required initial conditions for the test. They may be expressed in terms of parameters that must be set or in terms of test ordering in cases where the completion of one test establishes the necessary prerequisites for another test. The evaluator must determine that the prerequisites are complete and appropriate in that they will not bias the observed test results towards the expected test results.

1574 The test steps and expected results specify the actions and parameters to be applied to the interfaces as well as how the expected results should be verified and what they are. The evaluator must determine that the test steps and expected results are consistent with the functional specification and the high-level design. The tests must verify behaviour documented in these specifications. This means that each security functional behaviour characteristic explicitly described in the functional specification and high-level design should have tests and expected results to verify that behaviour.

1575 Although all of the TSF has to be tested by the developer, exhaustive specification testing of the interfaces is not required. The overall aim of this activity is to determine that each security function has been sufficiently tested against the behavioural claims in the functional specification and high-level design. The test procedures will provide insight as to how the security functions have been exercised by the developer during testing. The evaluator will use this information when developing additional tests to independently test the TOE.

8.9.2 Evaluation of coverage (ATE_COV.2)

8.9.2.1 Objectives

1576 The objective of this sub-activity is to determine whether the testing (as documented) is sufficient to establish that the TSF has been systematically tested against the functional specification.

8.9.2.2 Input

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test coverage analysis.

8.9.2.3 Evaluator actions

This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_COV.2.1E.

8.9.2.3.1 Action ATE_COV.2.1E

ATE_COV.2.1C

4:ATE_COV.2-1 The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the tests identified in the test documentation and the functional specification is accurate.

1577 Correspondence may take the form of a table or matrix. In some cases mapping may be sufficient to show test correspondence. In other cases a rationale (typically prose) may have to supplement the correspondence analysis provided by the developer.

1578 Figure 8.2 displays a conceptual framework of the correspondence between security functions described in the functional specification and the tests outlined in the test documentation used to test them. Tests may involve one or multiple security functions depending on the test dependencies or the overall goal of the test being performed.

1579 The identification of the tests and the security functions presented in the test coverage analysis has to be unambiguous. The test coverage analysis will allow the evaluator to trace the identified tests back to the test documentation and the particular security function being tested back to the functional specification.

4:ATE_COV.2-2 The evaluator ***shall examine*** the test plan to determine that the testing approach for each security function of the TSF is suitable to demonstrate the expected behaviour.

1580 Guidance on this work unit can be found in:

a) Application notes, Section 8.9.1.1, Understanding the expected behaviour of the TOE;

b) Application notes, Section 8.9.1.2, Testing vs. alternate approaches to verify the expected behaviour of a security function.

4:ATE_COV.2-3 The evaluator ***shall examine*** the test procedures to determine that the test prerequisites, test steps and expected result(s) adequately test each security function.

Interp Note : The following paragraph is changed as a result of Interpretation 074.

1581 Guidance on this work unit, as it pertains to the functional specification, can be found in:

a) Application notes, Section 8.9.1.3, Verifying the adequacy of tests.

ATE_COV.2.2C

4:ATE_COV.2-4 The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

1582 All security functions and interfaces that are described in the functional specification have to be present in the test coverage analysis and mapped to tests in order for completeness to be claimed, although exhaustive specification testing of interfaces is not required. As Figure 8.2 displays, all the security functions have tests attributed to them and therefore complete test coverage is depicted in this example. Incomplete coverage would be evident if a security function was identified in the test coverage analysis and no tests could be attributed to it.

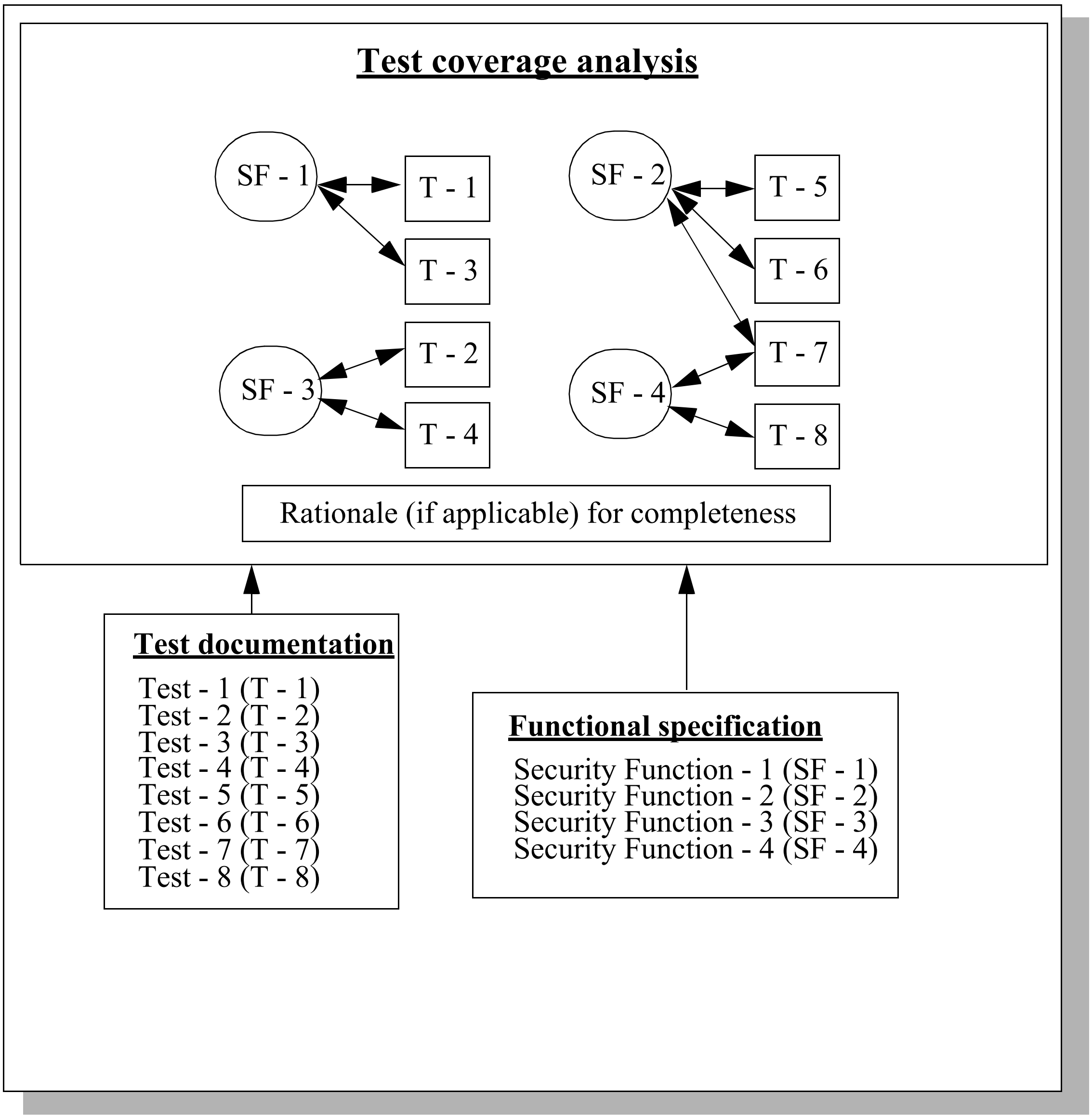

Figure 8.2 A conceptual framework of the test coverage analysis

8.9.3 Evaluation of depth (ATE_DPT.1)

8.9.3.1 Objectives

1583 The objective of this sub-activity is to determine whether the developer has tested the TSF against its high-level design.

8.9.3.2 Input

a) the ST;

b) the functional specification;

c) the high-level design;

d) the test documentation;

e) the depth of testing analysis.

8.9.3.3 Evaluator Actions

1584 This sub-activity comprises one CC part 3 evaluator action element:

a) ATE_DPT.1.1E.

8.9.3.3.1 Action ATE_DPT.1.1E

ATE_DPT.1.1C

4:ATE_DPT.1-1 The evaluator ***shall examine*** the depth of testing analysis for a mapping between the tests identified in the test documentation and the high-level design.

1585 The depth of testing analysis identifies all subsystems described in the high-level design and provides a mapping of the tests to these subsystems. Correspondence may take the form of a table or matrix. In some cases the mapping may be sufficient to show test correspondence. In other cases a rationale (typically prose) may have to supplement the mapping evidence provided by the developer.

1586 All design details specified in the high-level design that map to and satisfy TOE security requirements are subject to testing and hence, should be mapped to test documentation. Figure 8.3 displays a conceptual framework of the mapping between subsystems described in the high-level design and the tests outlined in the TOE's test documentation used to test them. Tests may involve one or multiple security functions depending on the test dependencies or the overall goal of the test being performed.

4:ATE_DPT.1-2 The evaluator ***shall examine*** the developer's test plan to determine that the testing approach for each security function of the TSF is suitable to demonstrate the expected behaviour.

1587 Guidance on this work unit can be found in:

a) Application notes, Section 8.9.1.1, Understanding the expected behaviour of the TOE;

b) Application notes, Section 8.9.1.2, Testing vs. alternate approaches to verify the expected behaviour of a security function.

1588 Testing of the TSF may be performed at the external interfaces, internal interfaces, or a combination of both. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the security functions. Specifically the evaluator determines whether testing at the internal interfaces for a security function is necessary or whether these internal interfaces can be adequately tested (albeit implicitly) by exercising the external interfaces. This determination is left to the evaluator, as is its justification.

4:ATE_DPT.1-3 The evaluator ***shall examine*** the test procedures to determine that the test pre-requisites, test steps and expected result(s) adequately test each security function.

Interp Note : The following paragraph is changed as a result of Interpretation 074.

1589 Guidance on this work unit, as it pertains to the high-level design, can be found in:

a) Application notes, Section 8.9.1.3, Verifying the adequacy of tests.

4:ATE_DPT.1-4 The evaluator ***shall check*** the depth of testing analysis to ensure that the TSF as defined in the high-level design is completely mapped to the tests in the test documentation.

1590 The depth of testing analysis provides a complete statement of correspondence between the high-level design and the test plan and procedures. All subsystems and internal interfaces described in the high-level design have to be present in the depth of testing analysis. All the subsystems and internal interfaces present in the depth of testing analysis must have tests mapped to them in order for completeness to be claimed. As Figure 8.3 displays, all the subsystems and internal interfaces have tests attributed to them and therefore complete depth of testing is depicted in this example. Incomplete coverage would be evident if a subsystem or internal interface was identified in the depth of testing analysis and no tests could be attributed to it.

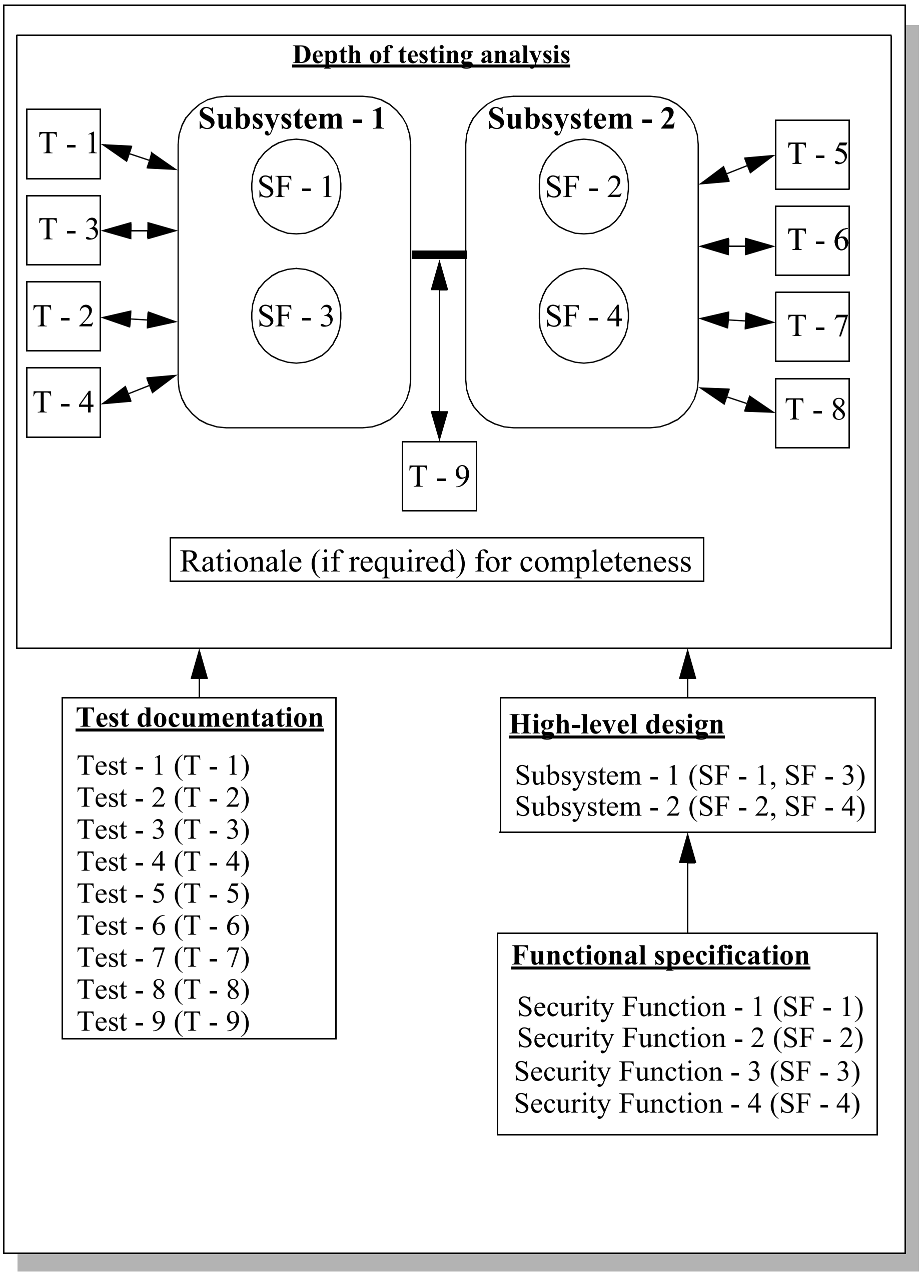

Figure 8.3 A conceptual framework of the depth of testing analysis

8.9.4 Evaluation of functional tests (ATE_FUN.1)

8.9.4.1 Objectives

1591 The objective of this sub-activity is to determine whether the developer's functional test documentation is sufficient to demonstrate that security functions perform as specified.

8.9.4.2 Application notes

1592 The extent to which the test documentation is required to cover the TSF is dependent upon the coverage assurance component.

1593 For the developer tests provided, the evaluator determines whether the tests are repeatable, and the extent to which the developer's tests can be used for the evaluator's independent testing effort. Any security function for which the developer's test results indicate that it may not perform as specified should be tested independently by the evaluator to determine whether or not it does.

8.9.4.3 Input

1594 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test procedures.

8.9.4.4 Evaluator actions

1595 This sub-activity comprises one CC Part 3 evaluator action element:

a) ATE_FUN.1.1E.

8.9.4.4.1 Action ATE_FUN.1.1E

ATE_FUN.1.1C

4:ATE_FUN.1-1 The evaluator **shall check** that the test documentation includes test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C

4:ATE_FUN.1-2 The evaluator ***shall check*** that the test plan identifies the security functions to be tested.

1596 One method that could be used to identify the security function to be tested is a reference to the appropriate part(s) of the functional specification that specifies the particular security function.

1597 The evaluator may wish to employ a sampling strategy when performing this work unit.

1598 For guidance on sampling see Annex B.2.

4:ATE_FUN.1-3 The evaluator ***shall examine*** the test plan to determine that it describes the goal of the tests performed.

1599 The test plan provides information about how the security functions are tested and the test configuration in which testing occurs.

1600 The evaluator may wish to employ a sampling strategy when performing this work unit.

1601 For guidance on sampling see Annex B.2.

4:ATE_FUN.1-4 The evaluator ***shall examine*** the test plan to determine that the TOE test configuration is consistent with the configuration identified for evaluation in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

1602 ~~The TOE used for testing should have the same unique reference as established by the ACM_CAP.4 sub-activity and the developer supplied test documentation.~~ The TOE referred to in the developer's test plan should have the same unique reference as established by the ACM_CAP.4 sub-activity.

1603 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator verifies that there are test configurations identified in the developer test documentation that are consistent with each evaluated configuration described in the ST.

1604 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

4:ATE_FUN.1-5 The evaluator ***shall examine*** the test plan to determine that it is consistent with the test procedure descriptions.

1605 The evaluator may wish to employ a sampling strategy when performing this work unit.

1606 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.3C

4:ATE_FUN.1-6 The evaluator ***shall check*** that the test procedure descriptions identify each security function behaviour to be tested.

1607 One method that may be used to identify the security function behaviour to be tested is a reference to the appropriate part(s) of the design specification that specifies the particular behaviour to be tested.

1608 The evaluator may wish to employ a sampling strategy when performing this work unit.

1609 For guidance on sampling see Annex B.2.

4:ATE_FUN.1-7 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to establish reproducible initial test conditions including ordering dependencies if any.

1610 Some steps may have to be performed to establish initial conditions. For example, user accounts need to be added before they can be deleted. An example of ordering dependencies on the results of other tests is the need to test the audit function before relying on it to produce audit records for another security mechanism such as access control. Another example of an ordering dependency would be where one test case generates a file of data to be used as input for another test case.

1611 The evaluator may wish to employ a sampling strategy when performing this work unit.

1612 For guidance on sampling see Annex B.2.

4:ATE_FUN.1-8 The evaluator ***shall examine*** the test procedure descriptions to determine that sufficient instructions are provided to have a reproducible means to stimulate the security functions and to observe their behaviour.

1613 Stimulus is usually provided to a security function externally through the TSFI. Once an input (stimulus) is provided to the TSFI, the behaviour of the security function can then be observed at the TSFI. Reproducibility is not assured unless the test procedures contain enough detail to unambiguously describe the stimulus and the behaviour expected as a result of this stimulus.

1614 The evaluator may wish to employ a sampling strategy when performing this work unit.

1615 For guidance on sampling see Annex B.2.

4:ATE_FUN.1-9 The evaluator ***shall examine*** the test procedure descriptions to determine that they are consistent with the test procedures.

1616 If the test procedure descriptions are the test procedures, then this work unit is not applicable and is therefore considered to be satisfied.

1617 The evaluator may wish to employ a sampling strategy when performing this work unit.

1618 For guidance on sampling see Annex B.2. For guidance on consistency analysis see Annex B.3.

ATE_FUN.1.4C

4:ATE_FUN.1-10 The evaluator ***shall examine*** the test documentation to determine that sufficient expected tests results are included.

1619 The expected test results are needed to determine whether or not a test has been successfully performed. Expected test results are sufficient if they are unambiguous and consistent with expected behaviour given the testing approach.

1620 The evaluator may wish to employ a sampling strategy when performing this work unit.

1621 For guidance on sampling see Annex B.2.

ATE_FUN.1.5C

4:ATE_FUN.1-11 The evaluator ***shall check*** that the expected test results in the test documentation are consistent with the actual test results provided.

1622 A comparison of the actual and expected test results provided by the developer will reveal any inconsistencies between the results.

1623 It may be that a direct comparison of actual results cannot be made until some data reduction or synthesis has been first performed. In such cases, the developer's test documentation should describe the process to reduce or synthesize the actual data.

1624 For example, the developer may need to test the contents of a message buffer after a network connection has occurred to determine the contents of the buffer. The message buffer will contain a binary number. This binary number would have to be converted to another form of data representation in order to make the test more meaningful. The conversion of this binary representation of data into a higher-level representation will have to be described by the developer in enough detail to allow an evaluator to perform the conversion process (i.e. synchronous or asynchronous transmission, number of stop bits, parity, etc.).

1625 It should be noted that the description of the process used to reduce or synthesize the actual data is used by the evaluator not to actually perform the necessary modification but to assess whether this process is correct. It is up to the developer to transform the expected test results into a format that allows an easy comparison with the actual test results.

1626 The evaluator may wish to employ a sampling strategy when performing this work unit.

1627 For guidance on sampling see Annex B.2.

1628 If the expected and actual test results for any test are not the same, then a demonstration of the correct operation of a security function has not been achieved. Such an occurrence will influence the evaluator's independent testing effort to include testing the implicated security function. The evaluator should also consider increasing the sample of evidence upon which this work unit is performed.

4:ATE_FUN.1-12 The evaluator ***shall report*** the developer testing effort, outlining the testing approach, configuration, depth and results.

1629 The developer testing information recorded in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing of the TOE by the developer. The intent of providing this information is to give a meaningful overview of the developer testing effort. It is not intended that the information regarding developer testing in the ETR be an exact reproduction of specific test steps or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the developer's testing approach, amount of testing performed, TOE test configurations, and the overall results of the developer testing.

1630 Information that would typically be found in the ETR section regarding the developer testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested;

b) testing approach. An account of the overall developer testing strategy employed;

c) amount of developer testing performed. A description on the extent of coverage and depth of developer testing;

d) testing results. A description of the overall developer testing results.

1631 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the developer testing effort.

8.9.5 Evaluation of independent testing (ATE_IND.2)

8.9.5.1 Objectives

1632 The goal of this activity is to determine, by independently testing a subset of the TSF, whether the TOE behaves as specified, and to gain confidence in the developer's test results by performing a sample of the developer's tests.

8.9.5.2 Input

1633 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the user guidance;

d) the administrator guidance;

e) the secure installation, generation, and start-up procedures;

f) the test documentation;

g) the test coverage analysis;

h) the depth of testing analysis;

i) the TOE suitable for testing.

8.9.5.3 Evaluator actions

1634 This sub-activity comprises three CC Part 3 evaluator action elements:

a) ATE_IND.2.1E;

b) ATE_IND.2.2E;

c) ATE_IND.2.3E.

8.9.5.3.1 Action ATE_IND.2.1E

ATE_IND.2.1C

4:ATE_IND.2-1 The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

Interp Note : The following two paragraphs are changed as a result of Interpretation 075.

1635 The TOE used for evaluator testing should have the same unique reference as established by the ACM_CAP.4 sub-activity ~~and the developer supplied test documentation~~.

1636 It is possible for the ST to specify more than one configuration for evaluation. The TOE may be composed of a number of distinct hardware and software implementations that need to be tested in accordance with the ST. The evaluator's TOE ~~verifies that there are~~ test configurations should be consistent with each evaluated configuration described in the ST.

1637 The evaluator should consider the assumptions about the security aspects of the TOE environment described in the ST that may apply to the test environment. There may be some assumptions in the ST that do not apply to the test environment. For example, an assumption about user clearances may not apply; however, an assumption about a single point of connection to a network would apply.

1638 If any test resources are used (e.g. meters, analysers) it will be the evaluator's responsibility to ensure that these resources are calibrated correctly.

4:ATE_IND.2-2 The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

1639 It is possible for the evaluator to determine the state of the TOE in a number of ways. For example, previous successful completion of the ADO_IGS.1 sub-activity will satisfy this work unit if the evaluator still has confidence that the TOE being used for testing was installed properly and is in a known state. If this is not the case, then the evaluator should follow the developer's procedures to install, generate and start up the TOE, using the supplied guidance only.

1640 If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed could satisfy work unit 4:ADO_IGS.1-2.

ATE_IND.2.2C

4:ATE_IND.2-3 The evaluator ***shall examine*** the set of resources provided by the developer to determine that they are equivalent to the set of resources used by the developer to functionally test the TSF.

1641 The resource set may include laboratory access and special test equipment, among others. Resources that are not identical to those used by the developer need to be equivalent in terms of any impact they may have on test results.

8.9.5.3.2 Action ATE_IND.2.2E

4:ATE_IND.2-4 The evaluator ***shall devise*** a test subset.

1642 The evaluator selects a test subset and testing strategy that is appropriate for the TOE. One extreme testing strategy would be to have the test subset contain as many security functions as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few security functions based on their perceived relevance and rigorously test these functions.

1643 Typically the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the security functional requirements identified in the ST using at least one test, but testing need not demonstrate exhaustive specification testing.

1644 The evaluator, when selecting the subset of the TSF to be tested, should consider the following factors:

a) The developer test evidence. The developer test evidence consists of: the test coverage analysis, the depth of testing analysis, and the test documentation. The developer test evidence will provide insight as to how the security functions have been exercised by the developer during testing. The evaluator applies this information when developing new tests to independently test the TOE. Specifically the evaluator should consider:

    1) augmentation of developer testing for specific security function(s). The evaluator may wish to perform more of the same type of tests by varying parameters to more rigorously test the security function.

    2) supplementation of developer testing strategy for specific security function(s). The evaluator may wish to vary the testing approach of a specific security function by testing it using another test strategy.

b) The number of security functions from which to draw upon for the test subset. Where the TOE includes only a small number of security functions, it may be practical to rigourously test all of the security functions. For TOEs with a large number of security functions this will not be cost-effective, and sampling is required.

c) Maintaining a balance of evaluation activities. The evaluator effort expended on the test activity should be commensurate with that expended on any other evaluation activity.

1645 The evaluator selects the security functions to compose the subset. This selection will depend on a number of factors, and consideration of these factors may also influence the choice of test subset size:

a) Rigour of developer testing of the security functions. All security functions identified in the functional specification had to have developer test evidence attributed to them as required by ATE_COV.2. Those security functions that the evaluator determines require additional testing should be included in the test subset.

b) Developer test results. If the results of developer tests cause the evaluator to doubt that a security function, or aspect thereof, operates as specified, then the evaluator should include such security functions in the test subset.

c) Known public domain weaknesses commonly associated with the type of TOE (e.g. operating system, firewall). Know public domain weaknesses associated with the type of TOE will influence the selection process of the test subset. The evaluator should include those security functions that address known public domain weaknesses for that type of TOE in the subset (know public domain weaknesses in this context does not refer to vulnerabilities as such but to inadequacies or problem areas that have been experienced with this particular type of TOE). If no such weaknesses are known, then a more general approach of selecting a broad range of security functions may be more appropriate.

d) Significance of security functions. Those security functions more significant than others in terms of the security objectives for the TOE should be included in the test subset.

e) SOF claims made in the ST. All security functions for which a specific SOF claim has been made should be included in the test subset.

f) Complexity of the security function. Complex security functions may require complex tests that impose onerous requirements on the developer or evaluator, which will not be conducive to cost-effective evaluations. Conversely, complex security functions are a likely area to find errors and are good candidates for the subset. The evaluator will need to strike a balance between these considerations.

g) Implicit testing. Testing some security functions may often implicitly test other security functions, and their inclusion in the subset may maximize the number of security functions tested (albeit implicitly). Certain interfaces will typically be used to provide a variety of security functionality, and will tend to be the target of an effective testing approach.

h) Types of interfaces to the TOE (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

i) Functions that are innovative or unusual. Where the TOE contains innovative or unusual security functions, which may feature strongly in marketing literature, these should be strong candidates for testing.

1646 This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

1647 For guidance on sampling see Annex B.2.

4:ATE_IND.2-5 The evaluator ***shall produce*** test documentation for the test subset that is sufficiently detailed to enable the tests to be reproducible.

1648 With an understanding of the expected behaviour of a security function, from the ST and the functional specification, the evaluator has to determine the most feasible way to test the function. Specifically the evaluator considers:

a) the approach that will be used, for instance, whether the security function will be tested at an external interface, at an internal interface using a test harness, or will an alternate test approach be employed (e.g. in exceptional circumstances, a code inspection);

b) the security function interface(s) that will be used to stimulate the security function and observe responses;

c) the initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

d) special test equipment that will be required to either stimulate a security function (e.g. packet generators) or make observations of a security function (e.g. network analysers).

1649 The evaluator may find it practical to test each security function using a series of test cases, where each test case will test a very specific aspect of expected behaviour.

1650 The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant design specification, and to the ST, if necessary.

4:ATE_IND.2-6 The evaluator ***shall conduct*** testing.

1651 The evaluator uses the test documentation developed as a basis for executing tests on the TOE. The test documentation is used as a basis for testing but this does not preclude the evaluator from performing additional ad hoc tests. The evaluator may devise new tests based on behaviour of the TOE discovered during testing. These new tests are recorded in the test documentation.

4:ATE_IND.2-7 The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the security function behaviour to be tested;

b) instructions to connect and setup all required test equipment as required to conduct the test;

c) instructions to establish all prerequisite test conditions;

d) instructions to stimulate the security function;

e) instructions for observing the behaviour of the security function;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE;

h) actual test results.

1652 The level of detail should be such that another evaluator could repeat the tests and obtain an equivalent result. While some specific details of the test results may be different (e.g. time and date fields in an audit record) the overall result should be identical.

1653 There may be instances when it is unnecessary to provide all the information presented in this work unit (e.g. the actual test results of a test may not require any analysis before a comparison between the expected results can be made). The determination to omit this information is left to the evaluator, as is the justification.

4:ATE_IND.2-8 The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

1654 Any differences in the actual and expected test results may indicate that the TOE does not perform as specified or that the evaluator test documentation may be incorrect. Unexpected actual results may require corrective maintenance to the TOE or test documentation and perhaps require re-running of impacted tests and modifying the test sample size and composition. This determination is left to the evaluator, as is its justification.

8.9.5.3.3 Action ATE_IND.2.3E

4:ATE_IND.2-9 The evaluator ***shall conduct*** testing using a sample of tests found in the developer test plan and procedures.

1655 The overall aim of this work unit is to perform a sufficient number of the developer tests to confirm the validity of the developer's test results. The evaluator has to decide on the size of the sample, and the developer tests that will compose the sample.

1656 Taking into consideration the overall evaluator effort for the entire tests activity, normally 20% of the developer's tests should be performed although this may vary according to the nature of the TOE, and the test evidence supplied.

1657 All the developer tests can be traced back to specific security function(s). Therefore, the factors to consider in the selection of the tests to compose the sample are similar to those listed for subset selection in work-unit ATE_IND.2-4. Additionally, the evaluator may wish to employ a random sampling method to select developer tests to include in the sample.

1658 For guidance on sampling see Annex B.2.

4:ATE_IND.2-10 The evaluator ***shall check*** that all the actual test results are consistent with the expected test results.

1659 Inconsistencies between the developer's expected test results and actual test results will compel the evaluator to resolve the discrepancies. Inconsistencies encountered by the evaluator could be resolved by a valid explanation and resolution of the inconsistencies by the developer.

1660 If a satisfactory explanation or resolution can not be reached, the evaluator's confidence in the developer's test results may be lessened and it may even be necessary for the evaluator to increase the sample size, to regain confidence in the developer testing. If the increase in sample size does not satisfy the evaluator's concerns, it may be necessary to repeat the entire set of developer's tests. Ultimately, to the extent that the TSF subset identified in work unit ATE_IND.2-4 is adequately tested, deficiencies with the developer's tests need to result in either corrective action to the developer's tests or in the production of new tests by the evaluator.

4:ATE_IND.2-11 The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

1661 The evaluator testing information reported in the ETR allows the evaluator to convey the overall testing approach and effort expended on the testing activity during the evaluation. The intent of providing this information is to give a meaningful overview of the testing effort. It is not intended that the information regarding testing in the ETR be an exact reproduction of specific test instructions or results of individual tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the testing approach chosen, amount of evaluator testing performed, amount of developer tests performed, TOE test configurations, and the overall results of the testing activity.

1662 Information that would typically be found in the ETR section regarding the evaluator testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested.

b) subset size chosen. The amount of security functions that were tested during the evaluation and a justification for the size.

c) selection criteria for the security functions that compose the subset. Brief statements about the factors considered when selecting security functions for inclusion in the subset.

d) security functions tested. A brief listing of the security functions that merited inclusion in the subset.

e) developer tests performed. The amount of developer tests performed and a brief description of the criteria used to select the tests.

f) verdict for the activity. The overall judgement on the results of testing during the evaluation.

This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the testing the evaluator performed during the evaluation.

8.10 Vulnerability assessment activity

1663 The purpose of the vulnerability assessment activity is to determine the existence and exploitability of flaws or weaknesses in the TOE in the intended environment. This determination is based upon analysis performed by the developer and the evaluator, and is supported by evaluator testing.

1664 The vulnerability assessment activity at EAL4 contains sub-activities related to the following components:

a) AVA_MSU.2;

b) AVA_SOF.1;

c) AVA_VLA.2.

8.10.1 Evaluation of misuse (AVA_MSU.2)

8.10.1.1 Objectives

1665 The objectives of this sub-activity are to determine whether the guidance is misleading, unreasonable or conflicting, whether secure procedures for all modes of operation have been addressed, and whether use of the guidance will facilitate prevention and detection of insecure TOE states.

8.10.1.2 Application notes

1666 The use of the term *guidance* in this sub-activity refers to the user guidance, the administrator guidance, and the secure installation, generation, and start-up procedures. Installation, generation, and start-up procedures here refers to all procedures the administrator is responsible to perform to progress the TOE from a delivered state to an operational state.

1667 This component includes a requirement for developer analysis that is not present in AVA_MSU.1. Validation of this analysis should not be used as a substitute for the evaluator's own examination of the guidance documentation, but should be used to provide evidence that the developer has also explicitly addressed the issue of misuse.

8.10.1.3 Input

1668 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the low-level design;

e) the subset of the implementation representation;

f) the TOE security policy model;

g) the user guidance;

h) the administrator guidance;

i) the secure installation, generation, and start-up procedures;

j) the misuse analysis of the guidance;

k) the test documentation;

l) the TOE suitable for testing.

8.10.1.4 Evaluator actions

1669 This sub-activity comprises four CC Part 3 evaluator action elements:

a) AVA_MSU.2.1E;

b) AVA_MSU.2.2E;

c) AVA_MSU.2.3E;

d) AVA_MSU.2.4E.

8.10.1.4.1 Action AVA_MSU.2.1E

AVA_MSU.2.1C

4:AVA_MSU.2-1 The evaluator ***shall examine*** the guidance and other evaluation evidence to determine that the guidance identifies all possible modes of operation of the TOE (including, if applicable, operation following failure or operational error), their consequences and implications for maintaining secure operation.

1670 Other evaluation evidence, particularly the functional specification and test documentation, provide an information source that the evaluator should use to determine that the guidance contains sufficient guidance information.

1671 The evaluator should focus on a single security function at a time, comparing the guidance for securely using the security function with other evaluation evidence, to determine that the guidance related to the security function is sufficient for the secure usage (i.e consistent with the TSP) of that security function. The evaluator should also consider the relationships between functions, searching for potential conflicts.

1672 AVA_MSU.2.2C

4:AVA_MSU.2-2 The evaluator ***shall examine*** the guidance to determine that it is clear and internally consistent.

1673 The guidance is unclear if it can reasonably be misconstrued by an administrator or user, and used in a way detrimental to the TOE, or to the security provided by the TOE.

1674 For guidance on consistency analysis see Annex B.3.

4:AVA_MSU.2-3 The evaluator ***shall examine*** the guidance and other evaluation evidence to determine that the guidance is complete and reasonable.

1675 The evaluator should apply familiarity with the TOE gained from performing other evaluation activities to determine that the guidance is complete.

1676 In particular, the evaluator should consider the functional specification and TOE summary specification. All security functions described in these documents should be described in the guidance as required to permit their secure administration and use. The evaluator may, as an aid, prepare an informal mapping between the guidance and these documents. Any omissions in this mapping may indicate incompleteness.

1677 The guidance is unreasonable if it makes demands on the TOE's usage or operational environment that are inconsistent with the ST or unduly onerous to maintain security.

1678 The evaluator should note that results gained during the performance of work units from the AGD_ADM sub-activity will provide useful input to this examination.

AVA_MSU.2.3C

4:AVA_MSU.2-4 The evaluator ***shall examine*** the guidance to determine that all assumptions about the intended environment are articulated.

1679 The evaluator analyses the assumptions about the intended TOE security environment of the ST and compares them with the guidance to ensure that all assumptions about the intended TOE security environment of the ST that are relevant to the administrator or user are described appropriately in the guidance.

AVA_MSU.2.4C

4:AVA_MSU.2-5 The evaluator ***shall examine*** the guidance to determine that all requirements for external security measures are articulated.

1680 The evaluator analyses the guidance to ensure that it lists all external procedural, physical, personnel and connectivity controls. The security objectives in the ST for the non-IT environment will indicate what is required.

AVA_MSU.2.5C

4:AVA_MSU.2-6 The evaluator ***shall examine*** the developer's analysis to determine that the developer has taken adequate measures to ensure that the guidance is complete.

1681 The developer analysis may comprise mappings from the ST or the functional specification to the guidance in order to demonstrate that the guidance is complete. Whatever evidence is provided by the developer to demonstrate completeness, the evaluator should assess the developer's analysis against any deficiencies found during the conduct of work units AVA_MSU.2-1 through AVA_MSU.2-5, and AVA_MSU.2-7.

8.10.1.4.2 Action AVA_MSU.2.2E

4:AVA_MSU.2-7 The evaluator ***shall perform*** all administrator and user (if applicable) procedures necessary to configure and install the TOE to determine that the TOE can be configured and used securely using only the supplied guidance.

1682 Configuration and installation requires the evaluator to advance the TOE from a deliverable state to the state in which it is operational and enforcing a TSP consistent with the security objectives specified in the ST.

1683 The evaluator should follow only the developer's procedures as documented in the user and administrator guidance that is normally supplied to the consumer of the TOE. Any difficulties encountered during such an exercise may be indicative of incomplete, unclear, inconsistent or unreasonable guidance.

1684 Note that work performed to satisfy this work unit may also contribute towards satisfying evaluator action ADO_IGS.1.2E.

4:AVA_MSU.2-8 The evaluator ***shall perform*** other security relevant procedures specified in the guidance to determine that the TOE can be configured and used securely using only supplied guidance.

1685 The evaluator should follow only the developer's procedures as documented in the user and administrator guidance that is normally supplied to the consumer of the TOE.

1686 The evaluator should employ sampling in carrying out this work unit. When choosing a sample the evaluator should consider:

a) the clarity of the guidance - any potential unclear guidance should be included in the sample;

b) guidance that will be used most often - infrequently used guidance should not normally be included in the sample;

c) complexity of the guidance - complex guidance should be included in the sample;

d) severity of error - procedures for which error imparts the greatest severity on security should be included in the sample;

e) the nature of the TOE - the guidance related to the normal or most likely use of the TOE should be included in the sample.

1687 For guidance on sampling see Annex B.2.

Interp Note : The following paragraph is removed as a result of Interpretation 133 (rev 1).

1688 ~~For guidance on consistency analysis see Annex B.3.~~

8.10.1.4.3 Action AVA_MSU.2.3E

4:AVA_MSU.2-9 The evaluator ***shall examine*** the guidance to determine that sufficient guidance is provided for the consumer to effectively administer and use the TOE's security functions, and to detect insecure states.

1689 TOEs may use a variety of ways to assist the consumer in effectively using the TOE securely. One TOE may employ functionality (features) to alert the consumer when the TOE is in an insecure state, whilst other TOEs may be delivered with enhanced guidance containing suggestions, hints, procedures, etc. on using the existing security features most effectively; for instance, guidance on using the audit feature as an aid for detecting insecure states.

1690 To arrive at a verdict for this work unit, the evaluator considers the TOE's functionality, its purpose and intended environment, and assumptions about its usage or users. The evaluator should arrive at the conclusion that, if the TOE can transition into an insecure state, there is reasonable expectation that use of the guidance would permit the insecure state to be detected in a timely manner. The potential for the TOE to enter into insecure states may be determined using the evaluation deliverables, such as the ST, the functional specification and the high-level design of the TSF.

8.10.1.4.4 Action AVA_MSU.2.4E

4:AVA_MSU.2-10 The evaluator ***shall examine*** the developer's analysis of the guidance to determine that guidance is provided for secure operation in all modes of operation of the TOE.

1691 The results of evaluation action AVA_MSU.2.1E should provide a basis with which to evaluate the developer's analysis. Having evaluated the potential for misuse of the guidance, the evaluator should be able to determine that the developer's misuse analysis meets the objectives of this sub-activity.

8.10.2 Evaluation of strength of TOE security functions (AVA_SOF.1)

8.10.2.1 Objectives

1692 The objectives of this sub-activity are to determine whether SOF claims are made in the ST for all probabilistic or permutational mechanisms and whether the developer's SOF claims made in the ST are supported by an analysis that is correct.

8.10.2.2 Application notes

1693 SOF analysis is performed on mechanisms that are probabilistic or permutational in nature, such as password mechanisms or biometrics. Although cryptographic mechanisms are also probabilistic in nature and are often described in terms of *strength*, AVA_SOF.1 is not applicable to cryptographic mechanisms. For such mechanisms, the evaluator should seek scheme guidance.

1694 Although SOF analysis is performed on the basis of individual mechanisms, the overall determination of SOF is based on functions. Where more than one probabilistic or permutational mechanism is employed to provide a security function, each distinct mechanism must be analysed. The manner in which these mechanisms combine to provide a security function will determine the overall SOF level for that function. The evaluator needs design information to understand how the mechanisms work together to provide a function, and a minimum level for such information is given by the dependency on ADV_HLD.1. The actual design information available to the evaluator is determined by the EAL, and the available information should be used to support the evaluator's analysis when required.

1695 For a discussion on SOF in relation to multiple TOE domains see Section 4.4.6.

8.10.2.3 Input

1696 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the low-level design;

e) the subset of the implementation representation;

f) the user guidance;

g) the administrator guidance;

h) the strength of TOE security functions analysis.

8.10.2.4 Evaluator actions

1697 This sub-activity comprises two CC Part 3 evaluator action elements:

a) AVA_SOF.1.1E;

b) AVA_SOF.1.2E.

8.10.2.4.1 Action AVA_SOF.1.1E

AVA_SOF.1.1C

4:AVA_SOF.1-1 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a SOF rating.

1698 If SOF claims are expressed solely as SOF metrics, then this work unit is not applicable and is therefore considered to be satisfied.

1699 A SOF rating is expressed as one of SOF-basic, SOF-medium or SOF-high, which are defined in terms of attack potential - refer to the CC Part 1 Glossary. A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational security mechanisms. However, individual mechanisms may have a SOF claim expressed as a rating that exceeds the overall SOF requirement.

1700 Guidance on determining the attack potential necessary to effect an attack and, hence, to determine SOF as a rating is in Annex B.8.

1701 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.2C

4:AVA_SOF.1-2 The evaluator ***shall check*** that the developer has provided a SOF analysis for each security mechanism for which there is a SOF claim in the ST expressed as a metric.

1702 If SOF claims are expressed solely as SOF ratings, then this work unit is not applicable and is therefore considered to be satisfied.

1703 A minimum overall SOF requirement expressed as a rating applies to all non-cryptographic, probabilistic or permutational mechanisms. However, individual mechanisms may have a SOF claim expressed as a metric that meets or exceeds the overall SOF requirement.

1704 The SOF analysis comprises a rationale justifying the SOF claim made in the ST.

AVA_SOF.1.1C and AVA_SOF.1.2C

4:AVA_SOF.1-3 The evaluator ***shall examine*** the SOF analysis to determine that any assertions or assumptions supporting the analysis are valid.

1705 For example, it may be a flawed assumption that a particular implementation of a pseudo-random number generator will possess the required entropy necessary to seed the security mechanism to which the SOF analysis is relevant.

1706 Assumptions supporting the SOF analysis should reflect the *worst case*, unless *worst case* is invalidated by the ST. Where a number of different possible scenarios exist, and these are dependent on the behaviour of the human user or attacker, the case that represents the lowest strength should be assumed unless, as previously stated, this case is invalid.

1707 For example, a strength claim based upon a maximum theoretical password space (i.e. all printable ASCII characters) would not be *worst case* because it is human behaviour to use natural language passwords, effectively reducing the password space and associated strength. However, such an assumption could be appropriate if the TOE used IT measures, identified in the ST, such as password filters to minimise the use of natural language passwords.

4:AVA_SOF.1-4 The evaluator ***shall examine*** the SOF analysis to determine that any algorithms, principles, properties and calculations supporting the analysis are correct.

1708 The nature of this work unit is highly dependent upon the type of mechanism being considered. Annex B.8 provides an example SOF analysis for an identification and authentication function that is implemented using a password mechanism; the analysis considers the maximum password space to ultimately arrive at a SOF rating. For biometrics, the analysis should consider resolution and other factors impacting the mechanism's susceptibility to spoofing.

1709 SOF expressed as a rating is based on the minimum attack potential required to defeat the security mechanism. The SOF ratings are defined in terms of attack potential in CC Part 1 Glossary.

1710 For guidance on attack potential see Annex B.8.

4:AVA_SOF.1-5 The evaluator ***shall examine*** the SOF analysis to determine that each SOF claim is met or exceeded.

1711 For guidance on the rating of SOF claims see Annex B.8.

4:AVA_SOF.1-6 The evaluator ***shall examine*** the SOF analysis to determine that all functions with a SOF claim meet the minimum strength level defined in the ST.

8.10.2.4.2 Action AVA_SOF.1.2E

4:AVA_SOF.1-7 The evaluator ***shall examine*** the functional specification, the high-level design, the low-level design, the user guidance and the administrator guidance to determine that all probabilistic or permutational mechanisms have a SOF claim.

1712 The identification by the developer of security functions that are realised by probabilistic or permutational mechanisms is verified during the ST evaluation. However, because the TOE summary specification may have been the only evidence available upon which to perform that activity, the identification of such mechanisms may be incomplete. Additional evaluation evidence required as input to this sub-activity may identify additional probabilistic or permutational mechanisms not already identified in the ST. If so, the ST will have to be updated appropriately to reflect the additional SOF claims and the developer will need to provide additional analysis that justifies the claims as input to evaluator action AVA_SOF.1.1E.

4:AVA_SOF.1-8 The evaluator ***shall examine*** the SOF claims to determine that they are correct.

1713 Where the SOF analysis includes assertions or assumptions (e.g. about how many authentication attempts are possible per minute), the evaluator should independently confirm that these are correct. This may be achieved through testing or through independent analysis.

8.10.3 Evaluation of vulnerability analysis (AVA_VLA.2)

8.10.3.1 Objectives

1714 The objective of this sub-activity is to determine whether the TOE, in its intended environment, has vulnerabilities exploitable by attackers possessing low attack potential.

8.10.3.2 Application notes

1715 The use of the term *guidance* in this sub-activity refers to the user guidance, the administrator guidance, and the secure installation, generation, and start-up procedures.

1716 The consideration of exploitable vulnerabilities will be determined by the security objectives and functional requirements in the ST. For example, if measures to prevent bypass of the security functions are not required in the ST (FPT_PHP, FPT_RVM and FPT_SEP are absent) then vulnerabilities based on bypass should not be considered.

1717 Vulnerabilities may be in the public domain, or not, and may require skill to exploit, or not. These two aspects are related, but are distinct. It should not be assumed that, simply because a vulnerability is in the public domain, it can be easily exploited.

1718 The following terms are used in the guidance with specific meaning:

a) Vulnerability - a weakness in the TOE that can be used to violate a security policy in some environment;

b) Vulnerability analysis - A systematic search for vulnerabilities in the TOE, and an assessment of those found to determine their relevance for the intended environment for the TOE;

c) Obvious vulnerability - a vulnerability that is open to exploitation that requires a minimum of understanding of the TOE, technical sophistication and resources;

d) Potential vulnerability - A vulnerability the existence of which is suspected (by virtue of a postulated attack path), but not confirmed, in the TOE;

e) Exploitable vulnerability - A vulnerability that can be exploited in the intended environment for the TOE;

f) Non-exploitable vulnerability - A vulnerability that cannot be exploited in the intended environment for the TOE;

g) Residual vulnerability - A non-exploitable vulnerability that could be exploited by an attacker with greater attack potential than is anticipated in the intended environment for the TOE;

h) Penetration testing - Testing carried out to determine the exploitability of identified TOE potential vulnerabilities in the intended environment for the TOE.

8.10.3.3 Input

1719 The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the high-level design;

d) the low-level design;

e) the subset of the implementation representation;

f) the TOE security policy model;

g) the user guidance;

h) the administrator guidance;

i) the secure installation, generation, and start-up procedures;

j) the vulnerability analysis;

k) the strength of function claims analysis;

l) the TOE suitable for testing.

1720 Other input for this sub-activity is:

a) current information regarding obvious vulnerabilities (e.g. from an overseer).

8.10.3.4 Evaluator actions

1721 This sub-activity comprises five CC Part 3 evaluator action elements:

a) AVA_VLA.2.1E;

b) AVA_VLA.2.2E;

c) AVA_VLA.2.3E;

d) AVA_VLA.2.4E;

e) AVA_VLA.2.5E.

8.10.3.4.1 Action AVA_VLA.2.1E

Interp Note : The following citation is changed as a result of Interpretation 051.

AVA_VLA.2.1C, ~~and~~ AVA_VLA.2.2C, AVA_VLA.2.3C and AVA_VLA.2.4C

4:AVA_VLA.2-1 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that the search for vulnerabilities has considered all relevant information.

1722 The developer's vulnerability analysis should cover the developer's search for vulnerabilities in at least all evaluation deliverables and public domain information sources.

Interp Note : The following paragraph is added as a result of Interpretation 031.

Information in the public domain is highly dynamic. Therefore, it is possible that new vulnerabilities are reported in the public domain between the time the developer performs the vulnerability analysis and the time that the evaluation is completed. The point at which monitoring of the public domain information ceases is an evaluation authority issue; therefore guidance and agreement should be sought from the evaluation authority.

4:AVA_VLA.2-2 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that each identified vulnerability is described and that a rationale is given for why it is not exploitable in the intended environment for the TOE.

1723 A vulnerability is termed non-exploitable if one or more of the following conditions exist:

a) security functions or measures in the (IT or non-IT) environment prevent exploitation of the vulnerability in the intended environment. For instance, restricting physical access to the TOE to authorised users only may effectively render a TOE's vulnerability to tampering unexploitable;

b) the vulnerability is exploitable but only by attackers possessing moderate or high attack potential. For instance, a vulnerability of a distributed TOE to session hijack attacks requires an attack potential beyond that of low. However, such vulnerabilities are reported in the ETR as residual vulnerabilities;

c) either the threat is not claimed to be countered or the violable organisational security policy is not claimed to be achieved by the ST. For instance, a firewall whose ST makes no availability policy claim and is vulnerable to TCP SYN attacks (an attack on a common Internet protocol that renders hosts incapable of servicing connection requests) should not fail this evaluator action on the basis of this vulnerability alone.

1724 For guidance on determining attack potential necessary to exploit a vulnerability see Annex B.8.

4:AVA_VLA.2-3 The evaluator ***shall examine*** the developer's vulnerability analysis to determine that it is consistent with the ST and the guidance.

1725 The developer's vulnerability analysis may address a vulnerability by suggesting specific configurations or settings for TOE functions. If such operating constraints are deemed to be effective and consistent with the ST, then all such configurations/ settings should be adequately described in the guidance so that they may be employed by the consumer.

8.10.3.4.2 Action AVA_VLA.2.2E

4:AVA_VLA.2-4 The evaluator ***shall devise*** penetration tests, building on the developer vulnerability analysis.

1726 The evaluator prepares for penetration testing:

a) as necessary to attempt to disprove the developer's analysis in cases where the developer's rationale for why a vulnerability is unexploitable is suspect in the opinion of the evaluator;

b) as necessary to determine the susceptibility of the TOE, in its intended environment, to a vulnerability not considered by the developer. The evaluator should have access to current information (e.g. from the overseer) regarding obvious public domain vulnerabilities that may not have been considered by the developer, and may also have identified potential vulnerabilities as a result of performing other evaluation activities.

1727 The evaluator is not expected to test for vulnerabilities (including those in the public domain) beyond those for which a low attack potential is required to effect an attack. In many cases, however, it will be necessary to carry out a test before the attack potential required can be determined. Where, as a result of evaluation expertise, the evaluator discovers a vulnerability that is beyond low attack potential, this is reported in the ETR as a residual vulnerability.

1728 With an understanding of the suspected vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically the evaluator considers:

a) the security function interfaces that will be used to stimulate the TSF and observe responses;

b) initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

c) special test equipment that will be required to either stimulate a security function or make observations of a security function.

1729 The evaluator will probably find it practical to carry out penetration testing using a series of test cases, where each test case will test for a specific vulnerability.

4:AVA_VLA.2-5 The evaluator ***shall produce*** penetration test documentation for the tests that build upon the developer vulnerability analysis, in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the vulnerability the TOE is being tested for;

b) instructions to connect and setup all required test equipment as required to conduct the penetration test;

c) instructions to establish all penetration test prerequisite initial conditions;

d) instructions to stimulate the TSF;

e) instructions for observing the behaviour of the TSF;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE.

1730 The intent of specifying this level of detail in the test documentation is to allow another evaluator to repeat the tests and obtain an equivalent result.

4:AVA_VLA.2-6 The evaluator ***shall conduct*** penetration testing, building on the developer vulnerability analysis.

1731 The evaluator uses the penetration test documentation resulting from work unit 4:AVA_VLA.2-4 as a basis for executing penetration tests on the TOE, but this does not preclude the evaluator from performing additional ad hoc penetration tests. If required, the evaluator may devise ad hoc tests as a result of information learned during penetration testing that, if performed by the evaluator, are to be recorded in the penetration test documentation. Such tests may be required to follow up unexpected results or observations, or to investigate potential vulnerabilities suggested to the evaluator during the pre-planned testing.

4:AVA_VLA.2-7 The evaluator ***shall record*** the actual results of the penetration tests.

1732 While some specific details of the actual test results may be different from those expected (e.g. time and date fields in an audit record) the overall result should be identical. Any differences should be justified.

4:AVA_VLA.2-8 The evaluator ***shall report*** in the ETR the evaluator penetration testing efforts, outlining the testing approach, configuration, depth and results.

1733 The penetration testing information reported in the ETR allows the evaluator to convey the overall penetration testing approach and effort expended on this sub-activity. The intent of providing this information is to give a meaningful overview of the evaluator’s penetration testing effort. It is not intended that the information regarding penetration testing in the ETR be an exact reproduction of specific test steps or results of individual penetration tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the penetration testing approach chosen, amount of penetration testing performed, TOE test configurations, and the overall results of the penetration testing activity.

1734 Information that would typically be found in the ETR section regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) security functions penetration tested. A brief listing of the security functions that were the focus of the penetration testing;

c) verdict for the sub-activity. The overall judgement on the results of penetration testing.

1735 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the penetration testing the evaluator performed during the evaluation.

8.10.3.4.3 Action AVA_VLA.2.3E

4:AVA_VLA.2-9 The evaluator ***shall examine*** all inputs to this sub-activity to determine possible security vulnerabilities not already addressed by the developer's vulnerability analysis.

1736 A flaw hypothesis methodology should be used whereby specifications and documentation for the TOE are analysed and then vulnerabilities in the TOE are hypothesised, or speculated. The list of hypothesised vulnerabilities is then prioritised on the basis of the estimated probability that a vulnerability exists and, assuming a vulnerability does exist, the attack potential required to exploit it, and on the extent of control or compromise it would provide. The prioritised list of potential vulnerabilities is used to direct penetration testing against the TOE.

1737 For guidance on determining attack potential necessary to exploit a vulnerability see Annex B.8.

1738 Vulnerabilities hypothesised as exploitable only by attackers possessing moderate or high attack potential do not result in a failure of this evaluator action. Where analysis supports the hypothesis, these need not be considered further as an input to penetration testing. However, such vulnerabilities are reported in the ETR as residual vulnerabilities.

1739 Vulnerabilities hypothesised exploitable by an attacker possessing a low attack potential, that do not result in a violation of the security objectives specified in the ST, do not result in a failure of this evaluator action. Where analysis supports the hypothesis, these need not be considered further as an input to penetration testing.

1740 Vulnerabilities hypothesised as potentially exploitable by an attacker possessing a low attack potential and resulting in a violation of the security objectives should be the highest priority potential vulnerabilities comprising the list used to direct penetration testing against the TOE.

1741 Subject to the threats being present in the intended environment, the evaluator's independent vulnerability analysis should consider generic vulnerabilities under each of the following headings:

a) generic vulnerabilities relevant for the type of TOE being evaluated, as may be supplied by the overseer;

b) bypassing;

c) tampering;

d) direct attacks;

e) misuse.

1742 Items b) - e) are now explained in greater detail.

Bypassing

1743 Bypassing includes any means by which an attacker could avoid security enforcement, by:

a) exploiting the capabilities of interfaces to the TOE, or of utilities which can interact with the TOE;

b) inheriting privileges or other capabilities that should otherwise be denied;

c) (where confidentiality is a concern) reading sensitive data stored or copied to inadequately protected areas.

1744 Each of the following should be considered (where relevant) in the evaluator's independent vulnerability analysis.

a) Attacks based on exploiting the capabilities of interfaces or utilities generally take advantage of the absence of the required security enforcement on those interfaces. For example, gaining access to functionality that is implemented at a lower level than that at which access control is enforced. Relevant items include:

    1) changing the predefined sequence of invocation of functions;

    2) executing an additional function;

    3) using a component in an unexpected context or for an unexpected purpose;

4) using implementation detail introduced in less abstract representations;

5) using the delay between time of access check and time of use.

b) Changing the predefined sequence of invocation of components should be considered where there is an expected order in which interfaces to the TOE (e.g. user commands) are called to perform some security function (e.g. opening a file for access and then reading data from it). If a security function is invoked on one of the TOE interfaces (e.g. an access control check), the evaluator should consider whether it is possible to bypass the security function by performing the call at a later point in the sequence or by missing it out altogether.

c) Executing an additional component (in the predefined sequence) is a similar form of attack to the one just described, but involves the calling of some other TOE interface at some point in the sequence. It can also involve attacks based on interception of sensitive data passed over a network by use of network traffic analysers (the additional component here being the network traffic analyser).

d) Using a component in an unexpected context or for an unexpected purpose includes using an unrelated TOE interface to bypass a security function by using it to achieve a purpose that it was not designed or intended to achieve. Covert channels are an example of this type of attack. The use of undocumented interfaces (which may be insecure) also falls into this category (these include undocumented support and help facilities).

e) Using implementation detail introduced in lower representations again includes the use of covert channels in which an attacker takes advantage of additional functions, resources or attributes that are introduced to the TOE as a consequence of the refinement process (e.g. use of a lock variable as a covert channel). Additional functionality may also include test harness code contained in software modules.

f) Using the delay between time of check and time of use includes scenarios where an access control check is made and access granted, and an attacker is subsequently able to create conditions in which, had they applied at the time the access check was made, would have caused the check to fail. An example would be a user creating a background process to read and send highly sensitive data to the user's terminal, and then logging out and logging back in again at a lower sensitivity level. If the background process is not terminated when the user logs off, the MAC checks would have been effectively bypassed.

g) Attacks based on inheriting privileges are generally based on illicitly acquiring the privileges or capabilities of some privileged component, usually by exiting from it in an uncontrolled or unexpected manner. Relevant items include:

1) executing data not intended to be executable, or making it executable;

2) generating unexpected input for a component;

3) invalidating assumptions and properties on which lower-level components rely.

h) Executing data not intended to be executable, or making it executable includes attacks involving viruses (e.g. putting executable code or commands in a file which are automatically executed when the file is edited or accessed, thus inheriting any privileges the owner of the file has).

i) Generating unexpected input for a component can have unexpected effects which an attacker could take advantage of. For example, if the TOE is an application implementing security functions that could be bypassed if a user gains access to the underlying operating system, it may be possible to gain such access following the login sequence by exploring the effect of hitting various control or escape sequences whilst a password is being authenticated.

j) Invalidating assumptions and properties on which lower level components rely includes attacks based on breaking out of the constraints of an application to gain access to an underlying operating system in order to bypass the security functions implemented by the application. In this case the assumption being invalidated is that it is not possible for a user of the application to gain such access. A similar attack can be envisaged if security functions are implemented by an application on an underlying database management system: again the security functions could be bypassed if an attacker can break out of the constraints of the application.

k) Attacks based on reading sensitive data stored in inadequately protected areas (applicable where confidentiality is a concern) include the following issues which should be considered as possible means of gaining access to sensitive data:

1) disk scavenging;

2) access to unprotected memory;

3) exploiting access to shared writable files or other shared resources (e.g. swap files);

4) Activating error recovery to determine what access users can obtain. For example, after a crash an automatic file recovery system may employ a lost and found directory for headerless files, which are on disc without labels. If the TOE implements mandatory access controls, it is important to investigate at what security level this directory is kept (e.g. at system high), and who has access to this directory.

Tampering

1745 Tampering includes any attack based on an attacker attempting to influence the behaviour of a security function or mechanism (i.e. corruption or de-activation), for example by:

a) accessing data on whose confidentiality or integrity the security function or mechanism relies;

b) forcing the TOE to cope with unusual or unexpected circumstances;

c) disabling or delaying security enforcement.

1746 Each of the following should be considered (where relevant) in the evaluator's independent vulnerability analysis.

a) Attacks based on accessing data on whose confidentiality or integrity the security function or mechanism include:

   1) reading, writing or modifying internal data directly or indirectly;

   2) using a component in an unexpected context or for an unexpected purpose;

   3) using interferences between components that are not visible at a higher level of abstraction.

b) Reading, writing or modifying internal data directly or indirectly includes the following types of attack which should be considered:

   1) reading 'secrets' stored internally, such as user passwords;

   2) spoofing internal data that security enforcing mechanisms rely upon;

   3) modifying environment variables (e.g. logical names), or data in configuration files or temporary files.

c) It may be possible to hoodwink a trusted process into modifying a protected file that it wouldn't normally access.

d) The evaluator should also consider the following 'dangerous features':

   1) source code resident on the TOE along with a compiler (for instance, it may be possible to modify the login source code);

   2) an interactive debugger and patch facility (for instance, it may be possible to modify the executable image);

   3) the possibility of making changes at device controller level, where file protection does not exist;

4) diagnostic code which exists in the source code and that may be optionally included;

5) developer's tools left in the TOE.

e) Using a component in an unexpected context or for an unexpected purpose includes (for example), where the TOE is an application built upon an operating system, users exploiting knowledge of a word processor package or other editor to modify their own command file (e.g. to acquire greater privileges).

f) Using interference between components which are not visible at a higher level of abstraction includes attacks exploiting shared access to resources, where modification of a resource by one component can influence the behaviour of another (trusted) component, e.g. at source code level, through the use of global data or indirect mechanisms such as shared memory or semaphores.

g) Attacks based on forcing the TOE to cope with unusual or unexpected circumstances should always be considered. Relevant items include:

1) generating unexpected input for a component;

2) invalidating assumptions and properties on which lower-level components rely.

h) Generating unexpected input for a component includes investigating the behaviour of the TOE when:

1) command input buffers overflow (possibly 'crashing the stack' or overwriting other storage, which an attacker may be able to take advantage of, or forcing a crash dump that may contain sensitive information such as clear-text passwords);

2) invalid commands or parameters are entered (including supplying a read-only parameter to an interface which expects to return data via that parameter);

3) an end-of-file marker (e.g. CTRL/Z or CTRL/D) or null character is inserted in an audit trail.

i) Invalidating assumptions and properties on which lower-level components rely includes attacks taking advantage of errors in the source code where the code assumes (explicitly or implicitly) that security relevant data is in a particular format or has a particular range of values. In these cases the evaluator should determine whether they can invalidate such assumptions by causing the data to be in a different format or to have different values, and if so whether this could confer advantage to an attacker.

j) The correct behaviour of the security functions may be dependent on assumptions that are invalidated under extreme circumstances where resource limits are reached or parameters reach their maximum value. The evaluator should consider (where practical) the behaviour of the TOE when these limits are reached, for example:

   1) changing dates (e.g. examining how the TOE behaves when a critical date threshold is passed);

   2) filling discs;

   3) exceeding the maximum number of users;

   4) filling the audit log;

   5) saturating security alarm queues at a console;

   6) overloading various parts of a multi-user TOE which relies heavily upon communications components;

   7) swamping a network, or individual hosts, with traffic;

   8) filling buffers or fields.

k) Attacks based on disabling or delaying security enforcement include the following items:

   1) using interrupts or scheduling functions to disrupt sequencing;

   2) disrupting concurrence;

   3) using interference between components which are not visible at a higher level of abstraction.

l) Using interrupts or scheduling functions to disrupt sequencing includes investigating the behaviour of the TOE when:

   1) a command is interrupted (with CTRL/C, CTRL/Y, etc.);

   2) a second interrupt is issued before the first is acknowledged.

m) The effects of terminating security critical processes (e.g. an audit daemon) should be explored. Similarly, it may be possible to delay the logging of audit records or the issuing or receipt of alarms such that it is of no use to an administrator (since the attack may already have succeeded).

n) Disrupting concurrence includes investigating the behaviour of the TOE when two or more subjects attempt simultaneous access. It may be that the TOE can cope with the interlocking required when two subjects attempt simultaneous access, but that the behaviour becomes less well defined in the presence of further subjects. For example, a critical security process could be put into a resource-wait state if two other processes are accessing a resource which it requires.

o) Using interference between components which are not visible at a higher level of abstraction may provide a means of delaying a time-critical trusted process.

Direct attacks

1747 Direct attack includes the identification of any penetration tests necessary to confirm or disprove the claimed minimum strength of functions. When identifying penetration tests under this heading, the evaluator should also be aware of the possibility of vulnerabilities existing as a result of security mechanisms being susceptible to direct attack.

Misuse

1748 Misuse includes the identification of any penetration tests necessary to confirm or disprove the misuse analysis. Issues to be considered include:

a) behaviour of the TOE when start-up, closedown or error recovery is activated;

b) behaviour of the TOE under extreme circumstances (sometimes termed overload or asymptotic behaviour), particularly where this could lead to the de-activation or disabling of a security enforcing function or mechanism;

c) any potential for unintentional misconfiguration or insecure use arising from attacks noted in the section on tampering above.

8.10.3.4.4 Action AVA_VLA.2.4E

4:AVA_VLA.2-10 The evaluator ***shall devise*** penetration tests, based on the independent vulnerability analysis.

1749 The evaluator prepares for penetration testing based on the prioritised list of vulnerabilities hypothesised in evaluator action AVA_VLA.2.3E.

1750 The evaluator is not expected to test for vulnerabilities beyond those for which a low attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a vulnerability that is exploitable only by an attacker with greater than low attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

1751 With an understanding of the suspected vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically the evaluator considers:

a) the security function interfaces that will be used to stimulate the TSF and observe responses;

b) initial conditions that will need to exist for the test (i.e. any particular objects or subjects that will need to exist and security attributes they will need to have);

c) special test equipment that will be required to either stimulate a security function or make observations of a security function.

1752 The evaluator will probably find it practical to carry out penetration test using a series of test cases, where each test case will test for a specific vulnerability.

4:AVA_VLA.2-11 The evaluator ***shall produce*** penetration test documentation for the tests based on the independent vulnerability analysis, in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the obvious vulnerability the TOE is being tested for;

b) instructions to connect and setup all required test equipment as required to conduct the penetration test;

c) instructions to establish all penetration test prerequisite initial conditions;

d) instructions to stimulate the TSF;

e) instructions for observing the behaviour of the TSF;

f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;

g) instructions to conclude the test and establish the necessary post-test state for the TOE.

1753 The intent of specifying this level of detail in the test documentation is to allow another evaluator to repeat the tests and obtain an equivalent result.

4:AVA_VLA.2-12 The evaluator ***shall conduct*** penetration testing, based on the independent vulnerability analysis.

1754 The evaluator uses the penetration test documentation resulting from work unit AVA_VLA.2-10 as a basis for executing penetration tests on the TOE, but this does not preclude the evaluator from performing additional ad hoc penetration tests. If required, the evaluator may devise new tests as a result of information learned during penetration testing that, if performed by the evaluator, are to be recorded in the penetration test documentation. Such tests may be required to follow up unexpected results or observations, or to investigate potential vulnerabilities suggested to the evaluator during the pre-planned testing.

1755 Should penetration testing show that a hypothesised vulnerability does not exist, then the evaluator should determine whether or not the evaluator's own analysis was incorrect, or if evaluation deliverables are incorrect or incomplete.

4:AVA_VLA.2-13 The evaluator ***shall record*** the actual results of the penetration tests.

1756 While some specific details of the actual test results may be different from those expected (e.g. time and date fields in an audit record) the overall result should be identical. Any differences should be justified.

4:AVA_VLA.2-14 The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

1757 The penetration testing information reported in the ETR allows the evaluator to convey the overall penetration testing approach and effort expended on this sub-activity. The intent of providing this information is to give a meaningful overview of the evaluator's penetration testing effort. It is not intended that the information regarding penetration testing in the ETR be an exact reproduction of specific test steps or results of individual penetration tests. The intention is to provide enough detail to allow other evaluators and overseers to gain some insight about the penetration testing approach chosen, amount of penetration testing performed, TOE test configurations, and the overall results of the penetration testing activity.

1758 Information that would typically be found in the ETR section regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) security functions penetration tested. A brief listing of the security functions that were the focus of the penetration testing;

c) verdict for the sub-activity. The overall judgement on the results of penetration testing.

1759 This list is by no means exhaustive and is only intended to provide some context as to the type of information that should be present in the ETR concerning the penetration testing the evaluator performed during the evaluation.

8.10.3.4.5 Action AVA_VLA.2.5E

4:AVA_VLA.2-15 The evaluator ***shall examine*** the results of all penetration testing and the conclusions of all vulnerability analysis to determine that the TOE, in its intended environment, is resistant to an attacker possessing a low attack potential.

1760 If the results reveal that the TOE, in its intended environment, has vulnerabilities exploitable by an attacker possessing less than a moderate attack potential, then this evaluator action fails.

4:AVA_VLA.2-16 The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. CEM activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the implicated security function(s), objective(s) not met, organisational security policy(ies) contravened and threat(s) realised;

c) a description;

d) whether it is exploitable in its intended environment or not (i.e. exploitable or residual);

e) identification of evaluation party (e.g. developer, evaluator) who identified it.

Annex A

Glossary

1761 This annex presents abbreviations, acronyms and vocabulary used by the CEM and does not include those already presented in the CC. This annex also presents the references used in the CEM.

A.1 Abbreviations and acronyms

1762 CEM Common Methodology for Information Technology Security Evaluation

1763 ETR Evaluation Technical Report

1764 OR Observation Report

A.2 Vocabulary

1765 Terms which are presented in bold-faced type are themselves defined in this section.

1766 Check:

to generate a **verdict** by a simple comparison. Evaluator expertise is not required. The statement that uses this verb describes what is mapped.

1767 Evaluation Deliverable:

any resource required from the sponsor or developer by the evaluator or overseer to perform one or more evaluation or evaluation oversight activities.

1768 Evaluation Evidence:

a tangible **evaluation deliverable**.

1769 Evaluation Technical Report:

a report that documents the **overall verdict** and its justification, produced by the evaluator and submitted to an overseer.

1770 Examine:

to generate a **verdict** by analysis using evaluator expertise. The statement that uses this verb identifies what is analysed and the properties for which it is analysed.

1771 Interpretation:

a clarification or amplification of a CC, CEM or **scheme** requirement.

1772 Methodology:

the system of principles, procedures and processes applied to IT security evaluations.

1773 Observation Report:

a report written by the evaluator requesting a clarification or identifying a problem during the evaluation.

1774 Overall Verdict:

a *pass* or *fail* statement issued by an evaluator with respect to the result of an evaluation.

1775 Oversight Verdict:

a statement issued by an overseer confirming or rejecting an **overall verdict** based on the results of evaluation oversight activities.

1776 Record:

to retain a written description of procedures, events, observations, insights and results in sufficient detail to enable the work performed during the evaluation to be reconstructed at a later time.

1777 Report:

to include evaluation results and supporting material in the **Evaluation Technical Report** or an **Observation Report**.

1778 Scheme:

set of rules, established by an evaluation authority, defining the evaluation environment, including criteria and **methodology** required to conduct IT security evaluations.

1779 Tracing:

a simple directional relation between two sets of entities, which shows which entities in the first set correspond to which entities in the second.

1780 Verdict:

a *pass*, *fail* or *inconclusive* statement issued by an evaluator with respect to a CC evaluator action element, assurance component, or class. Also see **overall verdict.**

A.3 References

CC Common Criteria for Information Technology Security Evaluation, Version 2.1, August 1999.

COD Concise Oxford Dictionary, Oxford University Press, Ninth edition, 1995.

IEEE IEEE Standard Glossary of Software Engineering Terminology, ANSI/IEEE STD 729-1983

Annex B

General evaluation guidance

B.1 Objectives

1781 The objective of this chapter is to cover general guidance used to provide technical evidence of evaluation results. The use of such general guidance helps in achieving objectivity, repeatability and reproducibility of the work performed by the evaluator.

B.2 Sampling

1782 This section provides general guidance on sampling. Specific and detailed information is given in those work units under the specific evaluator action elements where sampling has to be performed.

1783 Sampling is a defined procedure of an evaluator whereby some subset of a required set of evaluation evidence is examined and assumed to be representative for the entire set. It allows the evaluator to gain enough confidence in the correctness of particular evaluation evidence without analysing the whole evidence. The reason for sampling is to conserve resources while maintaining an adequate level of assurance. Sampling of the evidence can provide two possible outcomes:

a) The subset reveals no errors, allowing the evaluator to have some confidence that the entire set is correct.

b) The subset reveals errors and therefore the validity of the entire set is called into question. Even the resolution of all errors that were found may be insufficient to provide the evaluator the necessary confidence and as a result the evaluator may have to increase the size of the subset, or stop using sampling for this particular evidence.

1784 Sampling is a technique which can be used to reach a reliable conclusion if a set of evidence is relatively homogeneous in nature, e.g. if the evidence has been produced during a well defined process.

1785 The CC identifies the following evaluator action elements where sampling is explicitly acceptable:

a) ADV_RCR.3.2E: “The evaluator shall determine the accuracy of the proofs of correspondence by selectively verifying the formal analysis.”

b) ATE_IND.*.2E: “The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified”.

c) ATE_IND.2.3E: “The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.”

d) AVA_CCA.*.3E: “The evaluator shall selectively validate the covert channel analysis through testing.”

e) AVA_MSU.2.2E and AVA_MSU.3.2E: “The evaluator shall repeat all configuration and installation procedures, and other procedures selectively, to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.”

f) AMA_SIA.1.2E: “The evaluator shall check, by sampling, that the security impact analysis documents changes to an appropriate level of detail, together with appropriate justifications that assurance has been maintained in the current version of the TOE.”

1786 In addition ADV_IMP.1.1D requires that the developer provide the implementation representation for a subset of the TSF only. The sample of the subset should be selected in agreement with the evaluator. Provision of a sample of the implementation representation allows the evaluator to assess the presentation of the implementation representation itself and to sample the traceability evidence to gain assurance in the correspondence between the low-level design and the implementation representation.

1787 In addition to the sampling that the CC accepts, the CEM identifies the following actions where sampling is acceptable:

a) Action ACM_CAP.*.1E: “The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.”

   Here sampling is accepted for the content and presentation of evidence elements ACM_CAP.*.8C and ACM_CAP.*.9C for EAL3 and EAL4.

b) Action ATE_FUN.1.1E: “The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.”

   Here sampling is accepted for the content and presentation of evidence element ATE_FUN.1.3C, ATE_FUN.1.4C, and ATE_FUN.1.5C for EAL2, EAL3, and EAL4.

c) Action ALC_DVS.1.1E: “The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.”

   Here sampling is accepted for the content and presentation of evidence element ALC_DVS.1.2C for EAL3 and EAL4.

1788 Sampling in the cases identified in the CC, and in cases specifically covered in CEM work items, is recognised as a cost-effective approach to performing evaluator actions. Sampling in other areas is permitted only in exceptional cases, where performance of a particular activity in its entirety would require effort disproportionate to the other evaluation activities, and where this would not add correspondingly to assurance. In such cases a rationale for the use of sampling in that area will need to be made. Neither the fact that the TOE is large and complex, nor that it has many security functional requirements, is sufficient justification, since evaluations of large, complex TOEs can be expected to require more effort. Rather it is intended that this exception be limited to cases such as that where the TOE development approach yields large quantities of material for a particular CC requirement that would normally all need to be checked or examined, and where such an action would not be expected to raise assurance correspondingly.

1789 Sampling needs to be justified taking into account the possible impact on the security objectives and threats of the TOE. The impact depends on what might be missed as a result of sampling. Consideration also needs to be given to the nature of the evidence to be sampled, and the requirement not to diminish or ignore any security functions.

1790 It should be recognised that sampling of evidence directly related to the implementation of the TOE (e.g. developer test results) requires a different approach to sampling related to the determination of whether a process is being followed. In many cases the evaluator is required to determine that a process is being followed, and a sampling strategy is recommended. The approach here will differ from that taken when sampling a developer's test results. This is because the former case is concerned with ensuring that a process is in place, and the latter deals with determining correct implementation of the TOE. Typically, larger sample sizes should be analysed in cases related to the correct implementation of the TOE than would be necessary to ensure that a process is in place.

1791 The following principles should be followed whenever sampling is performed:

Interp Note : The following paragraph is changed as a result of Interpretation 120.

a) The sample size should be commensurate with the cost effectiveness of the evaluation and will depend on a number of TOE dependent factors (e.g. the size and complexity of the TOE, the amount of documentation), but a minimum size of 20% should be adopted as a norm for sampling material related to the TOE implementation. ~~Where sampling relates to gaining evidence that a process (e.g. visitor control or design review), a percentage figure is not appropriate, and the evaluator should sample sufficient information to gain reasonable confidence that the procedure is being followed. The sample size has to be justified.~~ Where sampling relates to gaining evidence that a process (e.g. visitor control or design review) is being followed, a percentage figure is not appropriate. The evaluator should sample sufficient information to gain reasonable confidence that the process is being followed, and justify the sample size.

b) The sample should be representative of all aspects relevant to the areas that are sampled. In particular, a selection should cover a variety of components, security functions, developer and operational sites (if more than one is involved) and hardware platform types (if more than one is involved).

c) The sponsor and developer should not be informed in advance of the exact composition of the sample, subject to ensuring timely delivery of the sample and supporting deliverable, e.g. test harnesses and equipment to the evaluator in accordance with the evaluation schedule.

d) The choice of the sample should be free from bias to the degree possible (one should not always choose the first or last item). Ideally the sample selection should be done by someone other than the evaluator.

1792 Errors found in the sample can be categorised as being either systematic or sporadic. If the error is systematic, the problem should be corrected and a complete new sample taken. If properly explained, sporadic errors might be solved without the need for a new sample, although the explanation should be confirmed. The evaluator should use judgement in determining whether to increase the sample size or use a different sample.

B.3 Consistency analysis

1793 This section provides general guidance on consistency analysis. Specific and detailed information is given in those work units under the specific evaluator action elements where a consistency analysis has to be performed.

1794 A consistency analysis is a defined procedure of an evaluator whereby a special part of an evaluation deliverable is itself analysed (internally consistent) or is compared with one or more other evaluation deliverables.

1795 The CC distinguishes between different kinds of consistency analysis:

a) The evaluator has to analyse the internal consistency of an evaluation deliverable. Examples are:

- ADV_FSP.1.2C: “The functional specification shall be internally consistent.”
- ADV_HLD.1.2C: “The high-level design shall be internally consistent.”
- ADV_IMP.1.2C: “The implementation representation shall be internally consistent.”
- ADV_LLD.1.2C: “The low-level design shall be internally consistent.”

While performing an internal consistency analysis the evaluator has to ensure that the deliverable provided does not include ambiguities. The evaluation deliverable should not include contradictory statements contained in different portions of the deliverable. For example, informal, semiformal, or formal presentations of the same evidence should agree with one another.

The evaluator should consider that parts of an evaluation deliverable may exist in separate documents (e.g. procedures for the secure installation, generation, and start-up may exist in three different documents).

b) The evaluator has to analyse that an evaluation deliverable is consistent with one or more other deliverables. Examples are:

- AGD_ADM.1.7C: “The administrator guidance shall be consistent with all other documentation supplied for evaluation.”
- AGD_USR.1.5C: “The user guidance shall be consistent with all other documentation supplied for evaluation.”

This consistency analysis requires the evaluator to verify that the descriptions of functions, security parameters, procedures and security-relevant events described in one document are consistent with those described in other documents supplied for evaluation. This means that the evaluator should consider possible inconsistencies with other sources of information. Examples include:

- inconsistencies with other guidelines on the use of security functions;
- inconsistencies with the ST (e.g. threats, secure usage assumptions, non-IT security objectives, or IT security functions);
- inconsistent use of security parameters with their description in the functional specification or low-level design;
- inconsistent description of security-relevant events with respect to information presented in the high-level or low-level design documents;
- conflicts of security enforcing functions with the informal TSP model.

c) The evaluator has to analyse both that an evaluation deliverable is internally consistent, and that an evaluation deliverable is consistent with other deliverables. An example is:

- AVA_MSU.1.2C: “The guidance documentation shall be complete, clear, consistent, and reasonable.”

Here it is required that guidance as a whole meet the requirement for consistency. Given that such guidance documentation may be contained in a single document, or in many separate documents, the requirement covers consistency across all guidance, within and between documents.

d) The evaluator has to check an analysis provided by the developer that is required to demonstrate consistency. Examples are:

- ADV_SPM.1.3C: “The TSP model shall include a rationale that demonstrates that it is consistent and complete with respect to all policies of the TSP that can be modelled.”

- ADV_SPM.1.4C: "The demonstration of correspondence between the TSP model and the functional specification shall show that all of the security functions in the functional specification are consistent and complete with respect to the TSP model."

In these cases it is the developer who has to present the evidence for consistency. However, the evaluator has to understand this analysis and has to confirm it, possibly even performing an independent analysis if necessary.

1796 The consistency analysis can be performed by examination of the evaluation deliverable(s). The evaluator should adopt a reasonable and structured approach to analysing the consistency of documents and may combine it with other activities, such as mapping or traceability, that are performed as part of other work units. The evaluator may be able to resolve any inconsistencies found by appealing to the formal description, if any. Similarly, use of semi-formal notations in deliverables, whilst not as precise as formal notation, can be used to reduce ambiguity in the deliverables.

1797 Ambiguity can arise explicitly from, for example, conflicting statements or implicitly when statements are not sufficiently precise. It should be noted that verbosity is not, in itself, sufficient grounds to assume a fail verdict against the consistency criteria.

1798 The consistency check of deliverables may highlight omissions that may require a rework of already performed work units. For example, the consistency check of the security objectives may identify an omission of one or more security requirements. In this case the evaluator should check the correspondence between the security objectives and the TSF.

B.4 Dependencies

1799 In general it is possible to perform the required evaluation activities, sub-activities, and actions in any order or in parallel. However, there are different kinds of dependencies which have to be considered by the evaluator. This section provides general guidance on dependencies between different activities, sub-activities, and actions.

B.4.1 Dependencies between activities

1800 For some cases the different assurance classes may recommend or even require a sequence for the related activities. A specific instance is the ST activity. The ST evaluation activity is started prior to any TOE evaluation activities since the ST provides the basis and context to perform them. However, a final verdict on the ST evaluation may not be possible until the TOE evaluation is complete, since changes to the ST may result from activity findings during the TOE evaluation.

B.4.2 Dependencies between sub-activities

1801 Dependencies identified between components in CC Part 3 have to be considered by the evaluator. An example for this kind of dependency is AVA_VLA.1. This component claims dependencies on ADV_FSP.1, ADV_HLD.1, AGD_ADM.1 and AGD_USR.1.

1802 A sub-activity can be assigned a pass verdict normally only if all those sub-activities are successfully completed on which it has a dependency. For example, a pass verdict on AVA_VLA.1 can normally only be assigned if the sub-activities related to ADV_FSP.1, ADV_HLD.1, AGD_ADM.1 and AGD_USR.1 are assigned a pass verdict too.

1803 So when determining whether a sub-activity will impact another sub-activity, the evaluator should consider whether this activity depends on potential evaluation results from any dependent sub-activities. Indeed, it may be the case that a dependent sub-activity will impact this sub-activity, requiring previously completed evaluator actions to be performed again.

1804 A significant dependency effect occurs in the case of evaluator-detected flaws. If a flaw is identified as a result of conducting one sub-activity, the assignment of a pass verdict to a dependent sub-activity may not be possible until all flaws related to the sub-activity upon which it depends are resolved.

B.4.3 Dependencies between actions

1805 It may be the case, that results which are generated by the evaluator during one action are used for performing another action. For example, actions for completeness and consistency cannot be completed until the checks for content and presentation have been completed. This means for example that the evaluator is recommended to evaluate the PP/ST rationale after evaluating the constituent parts of the PP/ST.

B.5 Site visits

1806 This section provides general guidance on site visits. Specific and detailed information is given in work units for those activities where site visits are performed:

a) ACM_AUT;

b) ACM_CAP.n (with n>2);

c) ADO_DEL;

d) ALC_DVS.

1807 A development site visit is a useful means whereby the evaluator determines whether procedures are being followed in a manner consistent with that described in the documentation.

1808 Reasons for visiting sites include:

a) to observe the use of the CM system as described in the CM plan;

b) to observe the practical application of delivery procedures;

c) to observe the application of security measures during development.

1809 During an evaluation it is often necessary that the evaluator will meet the developer more than once and it is a question of good planning to combine the site visit with another meeting to reduce costs. For example one might combine the site visits for configuration management, for the developer's security and for delivery. It may also be necessary to perform more than one site visit to the same site to allow the checking of all development phases. It should be considered that development could occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites.

1810 The first site visit should be scheduled early during the evaluation. In the case of an evaluation which starts during the development phase of the TOE, this will allow corrective actions to be taken, if necessary. In the case of an evaluation which starts after the development of the TOE, an early site visit could allow corrective measures to be put in place if serious deficiencies in the applied procedures emerge. This avoids unnecessary evaluation effort.

1811 Interviews are also a useful means of determining whether the written procedures reflect what is done. In conducting such interviews, the evaluator should aim to gain a deeper understanding of the analysed procedures at the development site, how they are used in practice and whether they are being applied as described in the provided evaluation evidence. Such interviews complement but do not replace the examination of evaluation evidence.

1812 To prepare for the site visit a checklist, based on the evaluation evidence provided should be generated by the evaluator. The results of the site visit should be recorded.

1813 Site visits may not be deemed necessary if e.g. the development site has recently been visited for another TOE evaluation or particular ISO 9000 procedures were confirmed as being followed. Other approaches to gain confidence should be considered that provide an equivalent level of assurance (e.g. to analyse evaluation evidence). Any decision not to make a visit should be determined in consultation with the overseer.

B.6 TOE boundary

1814 The identity of what is evaluated will appear in the ETR, on the certificate, in the ST, and on the list of evaluated products. Although *products* are typically bought and sold, evaluations are concerned with *TOE*s. In cases where the developer of the product is also the developer of the evaluation evidence (i.e. the sponsor), this distinction is unnecessary. But because these roles may be filled by different parties, the following were agreed as the basis of definitions used in the CEM, along with their interrelationships and effects upon evaluations and certification.

B.6.1 Product and system

1815 The *product* is the collection of hardware and/or software that is available for use. Some purveyors might bundle a collection of products (e.g. a word processor, spreadsheet, and graphics application) into yet another product (e.g. an office automation system). But, provided that it is available for use, either by the public, by other manufacturers, or by limited customers, the resulting collection is considered to be a product.

1816 A *system* consists of one or more products in a known operational environment. The main difference between a product evaluation and a system evaluation is that, for a system evaluation, the evaluator takes into account the actual environment instead of theorising a hypothetical one, as done for a product evaluation.

B.6.2 TOE

1817 The *TOE* is the entity that is evaluated as defined by the ST. While there are cases where a TOE makes up the entire product, this need not be the case. The TOE may be a product, a part of a product, a set of products, a unique technology never to be made into a product, or combinations of all of these, in a specific configuration or set of configurations. This specific configuration or set of configurations is called the *evaluated configuration*. The ST clearly describes the relation between the TOE and any associated products.

Interp Note : The following paragraph is added as a result of Interpretation 025.

This evaluated configuration is identified in sufficient detail to differentiate hardware included in the evaluated configuration from hardware that is not included in the evaluated configuration, though it might be available as part of the product upon which the TOE is based. This identification makes it apparent to potential customers what product must be purchased, and what configuration options must be used, in order for the TOE to run securely.

B.6.3 TSF

1818 The *TSF* is the collection of those functions within the TOE that enforce the security of the TOE as defined by the ST. There may be functions within the TOE that contribute nothing to the security of the TOE as defined by the ST; consequently, such functions would not be part of the TSF.

Interp Note : The following paragraph is added as a result of Interpretation 025.

The hardware portions of the TSF are described at a level of detail commensurate with the assurance requirements related to the relevant development documentation (functional specification, high-level design, low-level design) and the testing documentation. The level of hardware identification is determined by the impact that the hardware features have upon the security functions and assurances being claimed.

B.6.4 Evaluation

1819 An implicit assumption for all evaluations is that the TOE is (by definition) the product or system in its evaluated configuration; this assumption need not be explicitly included in the list of assumptions for the evaluation. The TOE undergoes the scrutiny of the evaluation: analysis is performed only within the evaluated configuration, testing is performed upon this evaluated configuration, exploitable vulnerabilities are identified in this evaluated configuration, and assumptions are relevant only in the evaluated configuration. The ease with which the TOE can exit this configuration is important, and must be considered where AVA_MSU is called up. This will look at the robustness of the TOE configuration, and the impact of any accidental or intentional deviations from it that may occur without detection.

1820 The following example provides three TOEs, all of which are based upon the same virtual private networking (VPN) firewall product, but which yield different evaluation results because of the differences in the STs.

1821 **1) A VPN-firewall which is configured in such a way that the VPN functionality is turned off. All threats in the ST are concerned with access to the safe network from the unsafe network.**

1822 The TOE is the VPN-firewall configured in such a way that the VPN functionality is turned off. If the administrator were to configure the firewall such that some or all VPN functions were enabled, the product would not be in an evaluated configuration; it would therefore be considered to be unevaluated, and so nothing could be stated about its security.

1823 **2) A VPN-firewall, where all threats in the ST are concerned with access to the safe network from the unsafe network.**

1824 The TOE is the entire VPN-firewall. The VPN functions are part of the TOE, so one of the things to be determined during the evaluation would be whether there are means to gain access to the safe network from the unsafe network through the VPN functions.

1825 **3) A VPN-firewall, where all threats in the ST are concerned with either access to the safe network from the unsafe network or confidentiality of traffic on the unsafe network.**

1826 The TOE is the entire VPN-firewall. The VPN functions are part of the TOE, so one of the things to be determined during the evaluation would be whether the VPN functions permit the realisation of any of the threats described in the ST.

B.6.5 Certification

1827 From the earlier paragraphs, it is clear that evaluating the same product with different STs leads to different TOEs with different TSFs. Consequently, the Certificates, ETR, the STs, and the entries in the Evaluated Products List will have to differ among the evaluations to be of any use to potential customers.

1828 Note that, for the above example of three different firewall evaluations, the apparent differences between these Certificates would be subtle, as the three VPN-firewalls would all lead to certificates identifying the TOE as:

> *The XYZ Firewall product, as described in the Evaluated Configuration identified in Security Target #ABC.*

1829 with a different identifier for each ST *ABC*.

1830 Therefore, the evaluator has to ensure that the ST adequately describes the TOE in terms of what functionality is within the scope of the evaluation. A clear explanation is vital because prospective customers of evaluated products will consult the STs of the products that they are considering to buy in order to determine which security functionality of those products have been evaluated.

B.7 Threats and FPT requirements

1831 The PP/ST author identifies threats (and from a threat perspective, there is no distinction made between the threat of a malicious user and that from an incorrect implementation exploitable through the external interface of the TSF) and uses these to determine the inclusion or exclusion of FPT_PHP, FPT_SEP, and/or FPT_RVM in the PP/ST. That is, all of these requirement families presuppose a threat to the TOE of physical tampering, user interference, or bypass:

a) The requirement for TSF protection is directly related to the statement of environment for the TOE. Where the threat of tampering or bypass is cited, either explicitly or implicitly, measures must be provided, either by the TOE or its environment, to address the threat.

b) The threat of tampering or bypass is typically indicated by the presence in the TOE environment of untrusted subjects (commonly human users), and where motivation exists to attack the assets that the TOE is intended to protect.

c) When assessing the statement of security requirements in the PP/ST, the evaluator determines the need for TSF protection to meet the security objectives, and where this need is established checks for the presence of functional requirements to meet it. Where the need for protection is identified, and no such protection is provided by the TOE or its environment, then a fail verdict will be assigned to the PP/ST evaluation sub-activity APE/ASE_REQ.

1832 There must be some form of protection for the TOE if it is to be able to enforce its security policy. After all, if the TSF is not protected from corruption, there is no guarantee that its policy enforcement functions will perform as expected.

1833 This protection can be provided in several ways. In an operating system, in which there are multiple users who have a rich (programming) interface to the TOE, the TSF must be able to protect itself. However, if the TOE is such that it has a limited interface, or a restricted usage, the necessary protection may be provided through means outside the TOE.

1834 It is the PP/ST author's responsibility to choose a combination of TOE security functions, assumptions about the IT environment, and other assumptions that provides for the needed self protection of the TOE security functions. It is the evaluator's responsibility to confirm that the necessary protection is provided. Depending on the TOE and the assumptions, the needed protection may demand functional security requirements from the FPT class; but there are circumstances under which it may not.

B.7.1 TOEs not necessarily requiring the FPT class

1835 It is conceivable that some TOEs (such as an embedded TOE with no user interface) would not be subject to these threats. A PP/ST for a TOE providing a rich user interface that includes these threats yet has no FPT_PHP, FPT_RVM, and FPT_SEP requirements is most likely an invalid PP/ST. The TOEs that may not need to include the FPT self-protection requirements may be divided into three types:

B.7.1.1 TOEs with a Limited User Interface

1836 A TOE that provides only a limited interface to the (untrusted) user already, by virtue of its limited interface, may provide sufficient constraints on the user's actions that even a malicious user may not be able to corrupt the TOE. For example, a device like a calculator, or a user authentication token, may have only a few possible input keys. The untrusted user interface to a communications device such as a router or guard is even more restricted: users can communicate only indirectly, typically through protocol data units or messages.

B.7.1.2 TOE enforcing no relevant Security Policies

1837 A TOE enforcing no access control or information flow control policies would presumably have no concern about a user accessing data of another user or of the TSF. In such a case, there would be little need for the separation of users that FPT_SEP implies. Similarly, if there are no perceived assets (such as IT resources) in need of protection (such as against denial of service), there may not be a need for FPT requirements.

B.7.1.3 Protection is provided by the Environment

1838 Protection of the TSF is often to be provided by the TOE environment, rather than the TOE itself (e.g. as in the case of an application running on a trusted operating system, where the application is the TOE). In such cases the evaluation will consider whether the environmental mechanisms provide the required protection. The protection measures themselves are assumed to operate correctly, but the manner in which they are applied to protect the TOE can influence the scope of the evaluation.

1839 For example, the privilege assigned by an operating system to object files within an application will determine the application's potential for violating the underlying operating system's TSP. It is possible to conceive of two implementations of the same application that make differing use of operating system protection measures, such that significantly different TSFs would be implied. Thus, even where the protection mechanisms are implemented by the TOE environment it remains necessary to examine the use made of those mechanisms before a determination of the TSF can be made.

B.7.2 Impact upon Assurance Families

1840 The inclusion/exclusion of the FPT self-protection requirements from the PP/ST will affect the following requirements:

B.7.2.1 ADV

1841 Where the threat of tampering or bypass does not exist, the evaluation will focus on correct operation of the TSF. This will include consideration of all functions within the TOE that contribute directly or indirectly to the enforcement of the TSP. Functions that fall into neither of these categories need not be examined (the presence of errors in the implementation of these functions that can interfere with the correct operation of the TSF will be established through testing of the TSF).

1842 Where self-protection functions have been claimed, the description of their implementation will identify the protection mechanisms, from which a determination of the TSF boundaries can be made. Identification of the TSF boundaries and interfaces, together with a determination of the efficacy of the TSF protection mechanisms claimed, will allow the evaluation to be limited in scope. This limitation will exclude functions outside the TSF, since these cannot interfere with correct TSF operation. In many cases, the TSF boundary will include some functions that do not contribute to the enforcement of the TSP, and these functions will need to be examined during the evaluation. Those functions that can be determined not to fall within the TSF need not be examined by the evaluator.

B.7.2.2 AVA_VLA

1843 Vulnerability analysis in the CC determines the impact of vulnerabilities on the operation of the TOE in its intended environment. If no threat of tampering or bypass is identified in the ST, then the search for vulnerabilities by the developer and evaluator, where required, should exclude consideration of such attacks.

B.7.2.3 ATE_IND

1844 The application notes for ATE_IND call for testing of obvious public domain weaknesses that may be applicable to the TOE. Such weaknesses that are based on the intent to tamper or bypass the TOE need only be considered where such a threat has been identified.

B.8 Strength of function and vulnerability analysis

1845 A comparison shows that there are important differences and important similarities between a strength of TOE security function analysis and a vulnerability analysis.

1846 An important similarity is based in their use of attack potential. For both analyses, the evaluator determines the minimum attack potential required by an attacker to effect an attack, and arrives at some conclusion about the TOE's resistance to attacks. Table B.1 and Table B.2 demonstrate and further describe the relationship between these analyses and attack potential.

Table B.1 Vulnerability analysis and attack potential

vulnerability component	TOE resistant to attacker with attack potential of:	Remaining vulnerabilities only exploitable by attacker with attack potential of:
VLA.4	high	Not applicable - successful attack beyond practicality
VLA.3	moderate	high
VLA.2	low	moderate

Table B.2 Strength of TOE security function and attack potential

SOF rating	adequate protection against attacker with attack potential	insufficient protection against attacker with attack potential
SOF - high	high	Not applicable - successful attack beyond practicality
SOF - medium	moderate	high
SOF - basic	low	moderate

1847 Important differences between these analyses are based in the nature of the TOE security function as well as in the nature of the attack. Strength of TOE security function analysis is only performed on probabilistic or permutational functions, except those which are based on cryptography. Furthermore, the analysis assumes that the probabilistic or permutational security function is implemented flawlessly and that the security function is used during attack within the limits of its design and implementation. As shown in Table B.2, a SOF rating then reflects the attack, described in terms of attack potential, against which the probabilistic or permutational security function is designed to protect.

1848 A vulnerability analysis applies to all non-cryptographic TOE security functions, including ones that are probabilistic or permutational in nature. Unlike a SOF analysis, no assumptions are made regarding the correctness of the security function's design and implementation; nor are constraints placed on the attack method or the attacker's interaction with the TOE - if an attack is possible, then it is to be considered during the vulnerability analysis. As shown in Table B.1, successful evaluation against a vulnerability assurance component reflects the level of threat, described in terms of attack potential, against which all TOE security functions are designed and implemented to protect.

1849 Common use of the notion of attack potential creates a link between SOF claims and vulnerability assessments, but this link should not be seen as creating a mandatory binding between the level of SOF claim and the assurance component selected from AVA_VLA. For example, the choice of AVA_VLA.2, which requires resistance to attackers with a low attack potential, does not restrict the choice of SOF rating to SOF-basic. Given that a vulnerability is inherently present in any probabilistic or permutational function, and that such functions are usually prominent aspects of a public interface (e.g. a password), a PP/ST author may require a higher level of resistance to attack at these points, and may select a higher SOF rating. A minimum claim of SOF-basic is required wherever components for AVA_SOF are claimed. The AVA_VLA component claimed imposes a floor on the SOF claim, and a SOF claim of SOF-basic should be seen as inconsistent with selection of AVA_VLA.3.

B.8.1 Attack potential

B.8.1.1 Application of attack potential

1850 Attack potential is a function of expertise, resources and motivation; each of these factors will be discussed. Attack potential is especially considered by the evaluator in two distinct ways during the ST evaluation and the vulnerability assessment activities. During the ST evaluation, the evaluator determines whether or not the choice of the assurance requirement components, in particular the components of the AVA class, are commensurate with the threat attack potential (see ASE_REQ.1.4C). Cases where the assurance is not commensurate may mean either that the evaluation will not provide sufficient assurance, or that the evaluation will be unnecessarily onerous. During the vulnerability assessment the evaluator is using attack potential as a means of determining the exploitability of identified vulnerabilities in the intended environment.

B.8.1.2 Treatment of motivation

1851 Motivation is an attack potential factor that can be used to describe several aspects related to the attacker and the assets the attacker desires. Firstly, motivation can imply the likelihood of an attack - one can infer from a threat described as highly motivated that an attack is imminent, or that no attack is anticipated from an un-motivated threat. However, except for the two extreme levels of motivation, it is difficult to derive a probability of an attack occurring from motivation.

1852 Secondly, motivation can imply the value of the asset, monetarily or otherwise, to the either the attacker or the asset holder. An asset of very high value is more likely to motivate an attack compared to an asset of little value. However, other than in a very general way, it is difficult to relate asset value to motivation because the value of an asset is subjective - it depends largely upon the value an asset holder places on it.

1853 Thirdly, motivation can imply the expertise and resources with which an attacker is willing to effect an attack. One can infer that a highly motivated attacker is likely to acquire sufficient expertise and resources to defeat the measures protecting an asset. Conversely, one can infer that an attacker with significant expertise and resources is not willing to effect an attack using them if the attacker's motivation is low.

1854 During the course of preparing for and conducting an evaluation, all three aspects of motivation are at some point considered. The first aspect, likelihood of attack, is what may inspire a developer to pursue an evaluation. If the developer believes that the attackers are sufficiently motivated to mount an attack, then an evaluation can provide assurance of the ability of the TOE to thwart the attacker's efforts. Where the intended environment is well defined, for example in a system evaluation, the level of motivation for an attack may be known, and will influence the selection of countermeasures.

1855 Considering the second aspect, an asset holder may believe that the value of the assets (however measured) is sufficient to motivate attack against them. Once an evaluation is deemed necessary, the attacker's motivation is considered to determine the methods of attack that may be attempted, as well as the expertise and resources used in those attacks. Once examined, the developer is able to choose the appropriate assurance level, in particular the AVA requirement components, commensurate with the attack potential for the threats. During the course of the evaluation, and in particular as a result of completing the vulnerability assessment activity, the evaluator determines whether or not the TOE, operating in its intended environment, is sufficient to thwart attackers with the identified expertise and resources.

B.8.2 Calculating attack potential

1856 This section examines the factors that determine attack potential, and provides some guidelines to help remove some of the subjectivity from this aspect of the evaluation process. This approach should be adopted unless the evaluator determines that it is inappropriate, in which case a rationale is required to justify the validity of the alternative approach.

B.8.2.1 Identification and exploitation

1857 For an attacker to exploit a vulnerability the vulnerability must first be identified, and then exploited. This may appear to be a trivial separation, but is an important one. To illustrate this, consider first a vulnerability that is uncovered following months of analysis by an expert, and a simple attack method published on the Internet. Compare this with a vulnerability that is well known, but requires enormous time and resource to exploit. Clearly factors such as time need to be treated differently in these cases.

1858 For SOF analysis, the issue of exploitation will normally be the most important, since vulnerabilities in probabilistic or permutational mechanisms will often be self evident. Note, however, that this may not always be the case. With cryptographic mechanisms, for example, knowledge of subtle vulnerabilities may considerably affect the effectiveness of a brute force attack. Knowledge that users of a system tend to choose first names as passwords will have a similar effect For vulnerability assessments above AVA_VLA.1, the initial identification of vulnerabilities will become a much more important consideration, since the existence of difficult to uncover vulnerabilities may be promulgated, often rendering exploitation trivial.

B.8.2.2 Factors to be considered

1859 The following factors should be considered during analysis of the attack potential required to exploit a vulnerability:

a) Identification

   1) Time taken to identify;

   2) Specialist technical expertise;

   3) Knowledge of the TOE design and operation;

   4) Access to the TOE;

   5) IT hardware/software or other equipment required for analysis.

b) Exploitation

   1) Time taken to exploit;

   2) Specialist technical expertise;

   3) Knowledge of the TOE design and operation;

   4) Access to the TOE;

   5) IT hardware/software or other equipment required for exploitation.

1860 In many cases these factors are not independent, but may be substituted for each other in varying degrees. For example, expertise or hardware/software may be a substitute for time. A discussion of these factors follows.

1861 *Time* is the time taken by an attacker to identify or exploit an attack on a continuous basis. For the purposes of this discussion *within minutes* means an attack can be identified or exploited in less than half an hour; *within hours* means an attack can succeed in less than a day; *within days* means an attack can succeed in less than a month, and *in months* means a successful attack requires at least a month.

1862 *Specialist expertise* refers to the level of generic knowledge of the application area or product type (e.g. Unix operation systems, Internet protocols). Identified levels are as follows:

a) *Experts* are familiar with the underlying algorithms, protocols, hardware, structures, etc. implemented in the product or system type and the principles and concepts of security employed;

b) *Proficient* persons are knowledgeable in that they are familiar with the security behaviour of the product or system type;

c) *Laymen* are unknowledgeable compared to experts or proficient persons, with no particular expertise.

1863 *Knowledge of the TOE* refers to specific expertise in relation to the TOE. This is distinct from generic expertise, but not unrelated to it. Identified levels are as follows:

a) *No information* about the TOE, other than its general purpose;

b) *Public information* concerning the TOE (e.g. as gained from user guides);

c) *Sensitive information* about the TOE (e.g. knowledge of internal design).

1864 Care should be taken here to distinguish information required to identify the vulnerability from the information required to exploit it, especially in the area of sensitive information. To require sensitive information for exploitation would be unusual.

1865 *Access to the TOE* is also an important consideration, and has a relationship to the time factor. Identification or exploitation of a vulnerability may require considerable amounts of access to a TOE that may increase the likelihood of detection. Some attacks may require considerable effort off-line, and only brief access to the TOE to exploit. Access may also need to be continuous, or over a number of sessions. For the purposes of this discussion *within minutes* means that access is required for less than half an hour; *within hours* means access is required for less than a day; *within days* means access is required for less than a month, and *in months* means access is required for at least a month. Where access to the TOE does not increase the likelihood of detection (e.g. a smartcard in the attacker's possession), this factor should be ignored.

1866 *IT hardware/software or other equipment* refers to the equipment is required to identify or exploit a vulnerability.

a) *Standard equipment* is equipment that is readily available to the attacker, either for the identification of a vulnerability or for an attack. This equipment may be a part of the TOE itself (e.g. a debugger in an operating system), or can be readily obtained (e.g. Internet downloads, or simple attack scripts).

b) *Specialised equipment* is not readily available to the attacker, but could be acquired without undue effort. This could include purchase of moderate amounts of equipment (e.g. protocol analyser), or development of more extensive attack scripts or programs.

c) *Bespoke equipment* is not readily available to the public as it may need to be specially produced (e.g. very sophisticated software), or because the equipment is so specialised that its distribution is controlled, possibly even restricted. Alternatively, the equipment may be very expensive. Use of hundreds of PCs linked across the Internet would fall into this category.

1867 *Specialist expertise* and *knowledge of the TOE* are concerned with the information required for persons to be able to attack a TOE. There is an implicit relationship between an attacker's expertise and the ability to effectively make use of equipment in an attack. The weaker the attacker's expertise, the lower the potential to use equipment. Likewise, the greater the expertise, the greater the potential for equipment to be used in the attack. Although implicit, this relationship between expertise and the use of equipment does not always apply, for instance, when environmental measures prevent an expert attacker's use of equipment, or when, through the efforts of others, attack tools requiring little expertise to effectively use are created and freely distributed (e.g. via the Internet).

B.8.2.3 An approach to calculation

1868 The above section identifies the factors to be considered. However, further guidance is required if evaluations are to be conducted on a standard basis. The following approach is provided to assist in this process. The numbers have been provided with the objective of achieving ratings that are consistent with the relevant evaluation levels.

1869 Table B.3 identifies the factors discussed in the previous section and associates numeric values with the two aspects of identifying and exploiting a vulnerability. When determining the attack potential for a given vulnerability, one value should be selected from each column for each factor (giving 10 values). When selecting values the intended environment for the TOE should be assumed. The 10 values are summed, giving a single value. This value is then checked using Table B.4 to determine the rating.

1870 Where a factor falls close to the boundary of a range the evaluator should consider use of an intermediate value to those in the table. For example, if access to the TOE is required for 1 hour in order to exploit the vulnerability, or if access is detectable very rapidly, then a value between 0 and 4 may be selected for that factor. The table is intended as a guide.

Table B.3 Calculation of attack potential

Factor	Range	Identifying value	Exploiting value
Elapsed Time	< 0.5 hour	0	0
	< 1 day	2	3
	< 1 month	3	5
	> 1 month	5	8
	Not practical	*	*
* Indicates that the attack path is not exploitable within a timescale that would be useful to an attacker. Any value of * indicates a High rating.			
Expertise	Layman	0	0
	Proficient	2	2
	Expert	5	4
Knowledge of TOE	None	0	0
	Public	2	2
	Sensitive	5	4
Access to TOE	< 0.5 hour, or access undetectable	0	0
	< 1 day	2	4
	< 1 month	3	6
	> 1 month	4	9
	Not practical	*	*
Equipment	None	0	0
	Standard	1	2
	Specialised	3	4
	Bespoke	5	6

1871 For a given vulnerability it may be necessary to make several passes through the

1872 In the case of a vulnerability that has been identified and is in the public domain, the identifying values should be selected for an attacker to uncover that vulnerability in the public domain, rather than to initially identify it.

1873 Table B.4 should then be used to obtain a rating for the vulnerability.

Table B.4 Rating of vulnerabilities

Range of values	Resistant to attacker with attack potential of:	SOF rating
<10	No rating	
10-17	Low	Basic
18-24	Moderate	Medium
>25	High	High

1874 An approach such as this cannot take account of every circumstance or factor, but should give a better indication of the level of resistance to attack required to achieve the standard ratings. Other factors, such as the reliance on unlikely chance occurrences, or the likelihood of detection before an attack can be completed, are not included in the basic model, but can be used by an evaluator as justification for a rating other than those that the basic model might indicate.

1875 In cases where, for example, a password mechanism is being rated, and the TOE implementation is such that only a very few attempts are permitted before the attack is curtailed, the strength rating becomes related almost entirely to the probability of a correct guess during those few attempts. Such curtailment measures would be viewed as part of the access control function, and whereas the password mechanism itself may receive, for example, only a SOF-medium rating, the access control function may be judged to be SOF-high.

1876 It should be noted that whereas a number of vulnerabilities rated individually may indicate a high resistance to attack, the presence of other vulnerabilities may alter the table values, such that the combination of vulnerabilities indicates that a lower overall rating is applicable. In other words, the presence of one vulnerability may make another one easier to exploit. Such an assessment should form part of the developer and evaluator vulnerability analysis.

B.8.3 Example strength of function analysis

1877 The SOF analysis for a hypothetical pass number mechanism is provided below.

1878 Information gleaned from the ST and design evidence reveals that identification and authentication provides the basis upon which to control access to network resources from widely distributed terminals. Physical access to the terminals is not controlled by any effective means. The duration of access to a terminal is not controlled by any effective means. Authorised users of the system choose their own pass numbers when initially authorized to use the system, and thereafter upon user request. The system places the following restrictions on the pass numbers selected by the user:

a) the pass number must be at least four and no greater than six digits long;

b) consecutive numerical sequences are disallowed (such as 7,6,5,4,3);

c) repeating digits is disallowed (each digit must be unique).

1879 Guidance provided to the users at the time of pass number selection is that pass numbers should be as random as possible and should not be affiliated with the user in some way - a date of birth, for instance.

1880 The pass number space is calculated as follows:

a) Patterns of human usage are an important considerations that can influence the approach to searching a password space, and thus affect SOF. Assuming the worst case scenario and the user chooses a number comprising only four digits, the number of pass number permutations assuming that each digit must be unique is:

$$7(8)(9)(10) = 5040$$

b) The number of possible increasing sequences is seven, as is the number of decreasing sequences. The pass number space after disallowing sequences is:

$$5040 - 14 = 5026$$

1881 Based on further information gleaned from the design evidence, the pass number mechanism is designed with a terminal locking feature. Upon the sixth failed authentication attempt the terminal is locked for one hour. The failed authentication count is reset after five minutes so that an attacker can at best attempt five pass number entries every five minutes, or 60 pass number entries every hour.

1882 On average, an attacker would have to enter 2513 pass numbers, over 2513 minutes, before entering the correct pass number. The average successful attack would, as a result, occur in slightly less than:

$$\frac{2513 min}{60 \frac{min}{hour}} \approx 42 hours$$

1883 Using the approach described in the previous section the identifying values would be the minimum from each category (total 0), since the existence of the vulnerability in such a function is clear. For exploitation, based on the above calculations, it is possible that a layman can defeat the mechanism within days (given access to the TOE), without the use of any equipment, and with no knowledge of the TOE, giving a value of 11. Given the resulting sum, 11, the attack potential required to effect a successful attack is determined to be at least moderate.

1884 The SOF ratings are defined in terms of attack potential in CC Part 1, Section 2.3, Glossary. Since a mechanism must be resistant to an attacker with low attack potential to claim SOF-basic, and since the pass number mechanism is resistant to an attacker with low attack potential, then this pass number mechanism rates, at best, SOF-basic.

B.9 Scheme responsibilities

1885 This CEM describes the minimum technical work that evaluations conducted under oversight (scheme) bodies must perform. However, it also recognises (both explicitly and implicitly) that there are activities or methods upon which mutual recognition of evaluation results do not rely. For the purposes of thoroughness and clarity, and to better delineate where the CEM ends and an individual scheme's methodology begins, the following matters are left up to the discretion of the schemes. Schemes may choose to provide the following, although they may choose to leave some unspecified. (Every effort has been made to ensure this list is complete; evaluators encountering a subject neither listed here nor addressed in the CEM should consult with their evaluation schemes to determine under whose auspices the subject falls.)

1886 The matters that schemes may choose to specify include:

a) what is required in ensuring that an evaluation was done sufficiently - every scheme has a means of verifying the work of its evaluators, whether by requiring the evaluators to present their findings to the oversight body, by requiring the oversight body to redo the evaluator’s work, or by some other means that assures the scheme that all evaluation bodies are adequate and comparable.

b) process for disposing of evaluation evidence upon completion of an evaluation;

c) any requirements for confidentiality (on the part of the evaluator and the non-disclosure of information obtained during evaluation);

d) the course of action to be taken if a problem is encountered during the evaluation (whether the evaluation continues once the problem is remedied, or the evaluation ends immediately and the remedied product must be re-submitted for evaluation);

e) any specific (natural) language in which documentation must be provided;

f) any recorded evidence that must be submitted in the ETR - this CEM specifies the minimum to be reported in an ETR; however, individual schemes may require additional information to be included;

g) any additional reports (other than the ETR) required from the evaluators - for example, testing reports;

h) any specific ORs that may be required by the scheme, including the structure, recipients, etc. of any such ORs;

i) any specific content structure of any written report as a result from an ST evaluation - a scheme may have a specific format for all of its reports detailing results of an evaluation, be it the evaluation of a TOE or of an ST;

j) any additional PP/ST identification information required;

k) any activities to determine the suitability of explicitly-stated requirements in an ST;

l) any requirements for provision of evaluator evidence to support re-evaluation and re-use of evidence;

m) any specific handling of scheme identifiers, logos, trademarks, etc.;

n) any specific guidance in dealing with cryptography;

o) handling and application of scheme, national and international interpretations;

p) a list or characterisations of suitable alternative approaches to testing where testing is infeasible;

q) the mechanism by which an overseer can determine what steps an evaluator took while testing;

r) preferred test approach (if any): at internal interface or at external interface;

s) a list or characterisation of acceptable means of conducting the evaluator's vulnerability analysis (e.g. flaw hypothesis methodology);

t) information regarding any vulnerabilities and weaknesses to be considered;

~~Annex C~~

Interp Note : This annex is deleted. Comments on the CEM are to be submitted through the RI mechanism.

~~Providing CEM observation reports~~

~~C.1 Introduction~~

1887 ~~The Common Evaluation Methodology Editorial Board (CEMEB) provides this document to their sponsoring organisations for use within the IT security evaluation community. However, it recognises that this use may motivate observations and/or comments on the document for consideration in future versions.~~

1888 ~~This annex details a mechanism by which to comment on the CEM. This mechanism consists of a report format, the CEM Observation Report (CEMOR), to be used to articulate an observation. Any observations should be submitted through the sponsoring organisations listed in the Foreword of the document.~~

1889 ~~Any comments should be submitted in the CEMOR format provided. This will allow the CEMEB to process all comments in a common and methodical way. All reviewers should include, where possible, substitution text or a clear resolution for any of the conceptual problems, inconsistencies or technical difficulties identified.~~

~~C.2 Format of a CEMOR~~

1890 ~~A CEMOR shall contain all of the following fields, although one or more fields may be empty. Each field shall begin with the ASCII character "$", followed by an arabic number, followed by the ASCII character ":"~~

~~$1: Originator's name~~

1891 ~~Full name of the originator.~~

~~$2: Originator organisation~~

1892 ~~The originator's organisation/affiliation.~~

~~$3: Return address~~

1893 ~~Electronic mail or other address to acknowledge receipt of the CEMOR and request clarification, if necessary.~~

~~$4: Date~~

1894 ~~Submission date of observation YY/MM/DD.~~

~~$5: Originator's CEMOR identifier~~

1895 ~~This unique identifier is assigned to the CEMOR by the originator.~~

~~$6: Observation type~~

1896 ~~Possible types are "Editorial", "Technical", "Programmatic" or "Other".~~

~~$7: Title of the CEMOR~~

1897 ~~A short descriptive title for this CEMOR.~~

~~$8: CEM document reference~~

1898 ~~The single reference to the affected area of the CEM. This field shall identify the CEM version number, part number and Section number. Additionally, a paragraph number (or, if no paragraph number is relevant, the work unit, table or figure number) shall also be identified in this field.~~

~~$9: Statement of observation~~

1899 ~~Comprehensive description of the observation. There is no restriction regarding the length of this field. However, it shall contain text only; no figures or tables other than what can be achieved within the realm of ASCII shall be used.~~

~~$10: Suggested solution(s)~~

1900 ~~Proposed solution(s) for addressing the observation.~~

~~$$ End of CEMOR~~

1901 ~~Required to mark the end of CEMOR relevant information.~~

~~C.2.1 Example observation~~

~~$1: Pat Smith~~

~~$2: CC Evals Laboratory~~

~~$3: psmith@cclab~~

~~$4: 1999/11/10~~

~~$5: CEMOR.psmith.comment.1~~

~~$6: Technical~~

~~$7: Inconclusive verdict is not a verdict~~

~~$8: CEM v1.0, Part 2, Section 1.4, paragraph 28b~~

~~$9: A verdict should be something that is the result of analysis. If a verdict is not yet reached, it should be called something other than a verdict. An inconclusive verdict could imply that the work was completed but questions remained (i.e., the evaluator did not know whether it passed or failed.)~~

~~$10: Change the CEM to have two verdicts: pass and fail. Before a verdict is reached should just be denoting as 'awaiting verdict.'~~

~~$$~~

1902 ~~Several CEMORs may be combined into a single submission. If this is done, fields $1 through $4 need appear only once at the beginning. For each CEMOR submitted, Fields $5 through $10 would appear next. The $$ shall appear following the last CEMOR.~~

Annex X

Interp Note : This annex was created to provide the methodology for ALC_FLR. This family is not part of any EAL, so it has been appended here at the end of the annotated CEM.

Flaw remediation sub-activities

0.1 Evaluation of flaw remediation (ALC_FLR.1)

0.1.1 Objectives

1 The objective of this sub-activity is to determine whether the developer has established flaw remediation procedures that describe the tracking of security flaws, the identification of corrective actions, and the distribution of corrective action information to TOE users.

0.1.2 Input

2 The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation.

0.1.3 Evaluator actions

3 This sub-activity comprises one CC Part 3 evaluator action element:

a) ALC_FLR.1.1E.

0.1.3.1 Action ALC_FLR.1.1E

ALC_FLR.1.1C - *The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.*

ALC_FLR.1-1 **The evaluator shall examine the flaw remediation procedures documentation to determine that it describes the procedures used to track all reported security flaws in each release of the TOE.**

4 The procedures describe the actions that are taken by the developer from the time each suspected security flaw is reported to the time that it is resolved. This includes the flaw's entire timeframe, from initial detection through ascertaining the flaw is a security flaw, to resolution of the security flaw.

5 If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the ALC_FLR requirements) for the flaw remediation procedures to track it further; only that there be an explanation of why the flaw is not security-relevant.

6 While these requirements do not mandate that there be a publicised means for TOE users to report security flaws, they do mandate that all security flaws that are reported be tracked. That is, a reported security flaw cannot be ignored simply because it comes from outside the developer's organisation.

ALC_FLR.1.2C - *The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.*

ALC_FLR.1-2 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would produce a description of each security flaw in terms of its nature and effects.

7 The procedures identify the actions that are taken by the developer to describe the nature and effects of each security flaw in sufficient detail to be able to reproduce it. The description of the nature of a security flaw addresses whether it is an error in the documentation, a flaw in the design of the TSF, a flaw in the implementation of the TSF, etc. The description of the security flaw's effects identifies the portions of the TSF that are affected and how those portions are affected. For example, a security flaw in the implementation might be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password 'BACKDOOR'.

ALC_FLR.1-3 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would identify the status of finding a correction to each security flaw.

8 The flaw remediation procedures identify the different stages of security flaws. This differentiation includes at least: suspected security flaws that have been reported, suspected security flaws that have been confirmed to be security flaws, and security flaws whose solutions have been implemented. It is permissible that additional stages (e.g. flaws that have been reported but not yet investigated, flaws that are under investigation, security flaws for which a solution has been found but not yet implemented) be included.

ALC_FLR.1.3C - *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

ALC_FLR.1-4 The evaluator shall check the flaw remediation procedures to determine that the application of these procedures would identify the corrective action for each security flaw.

9 *Corrective action* may consist of a repair to the hardware, firmware, or software portions of the TOE, a modification of TOE guidance, or both. Corrective action that constitutes modifications to TOE guidance (e.g. details of procedural measures to be taken to obviate the security flaw) includes both those measures serving as only an interim solution (until the repair is issued) as well as those

serving as a permanent solution (where it is determined that the procedural measure is the best solution).

10 If the source of the security flaw is a documentation error, the corrective action consists of an update of the affected TOE guidance. If the corrective action is a procedural measure, this measure will include an update made to the affected TOE guidance to reflect these corrective procedures.

ALC_FLR.1.4C - *The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to the TOE users.*

ALC_FLR.1-5 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes a means of providing the TOE users with the necessary information on each security flaw.

11 The *necessary information* about each security flaw consists of its description (not necessarily at the same level of detail as that provided as part of work unit ALC_FLR.1-2), the prescribed corrective action, and any associated guidance on implementing the correction.

12 TOE users may be provided such information, correction, and documentation updates in any of several ways, such as their posting to a website, their being sent to TOE users, or arrangements made for the developer to install the correction. In cases where the means of providing this information requires action to be initiated by the TOE user, the evaluator examines any TOE guidance to ensure that it contains instructions for retrieving the information.

13 The only metric for assessing the adequacy of the method used for providing the information, corrections and guidance is that there be a reasonable expectation that TOE users can obtain or receive it. For example, consider the method of dissemination where the requisite data is posted to a website for one month, and the TOE users know that this will happen and when this will happen. This may not be especially reasonable or effective (as, say, a permanent posting to the website), yet it is feasible that the TOE user could obtain the necessary information. On the other hand, if the information were posted to the website for only one hour, yet TOE users had no way of knowing this or when it would be posted, it is infeasible that they would ever get the necessary information.

0.2 Evaluation of flaw remediation (ALC_FLR.2)

0.2.1 Objectives

14 The objective of this sub-activity is to determine whether the developer has established flaw remediation procedures that describe the tracking of security flaws, the identification of corrective actions, and the distribution of corrective action information to TOE users. Additionally, this sub-activity determines whether the developer's procedures provide for the corrections of security flaws, for the receipt of flaw reports from TOE users, and for assurance that the corrections introduce no new security flaws.

15 In order for the developer to be able to act appropriately upon security flaw reports from TOE users, TOE users need to understand how to submit security flaw reports to the developer, and developers need to know how to receive these reports. Flaw remediation guidance addressed to the TOE user ensures that TOE users are aware of how to communicate with the developer; flaw remediation procedures describe the developer's role is such communication

0.2.2 Input

16 The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation;

b) flaw remediation guidance documentation.

0.2.3 Evaluator actions

17 This sub-activity comprises one CC Part 3 evaluator action element:

a) ALC_FLR.2.1E.

0.2.3.1 Action ALC_FLR.2.1E

ALC_FLR.2.1C - *The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.*

ALC_FLR.2-1 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes the procedures used to track all reported security flaws in each release of the TOE.

18 The procedures describe the actions that are taken by the developer from the time each suspected security flaw is reported to the time that it is resolved. This includes the flaw's entire timeframe, from initial detection through ascertaining the flaw is a security flaw, to resolution of the security flaw.

19 If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the ALC_FLR requirements) for the flaw remediation procedures to

track it further; only that there be an explanation of why the flaw is not security-relevant.

ALC_FLR.2.2C - *The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.*

ALC_FLR.2-2 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would produce a description of each security flaw in terms of its nature and effects.

20 The procedures identify the actions that are taken by the developer to describe the nature and effects of each security flaw in sufficient detail to be able to reproduce it. The description of the nature of a security flaw addresses whether it is an error in the documentation, a flaw in the design of the TSF, a flaw in the implementation of the TSF, etc. The description of the security flaw's effects identifies the portions of the TSF that are affected and how those portions are affected. For example, a security flaw in the implementation might be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password 'BACKDOOR'.

ALC_FLR.2-3 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would identify the status of finding a correction to each security flaw.

21 The flaw remediation procedures identify the different stages of security flaws. This differentiation includes at least: suspected security flaws that have been reported, suspected security flaws that have been confirmed to be security flaws, and security flaws whose solutions have been implemented. It is permissible that additional stages (e.g. flaws that have been reported but not yet investigated, flaws that are under investigation, security flaws for which a solution has been found but not yet implemented) be included.

ALC_FLR.2.3C - *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

ALC_FLR.2-4 The evaluator shall check the flaw remediation procedures to determine that the application of these procedures would identify the corrective action for each security flaw.

22 *Corrective action* may consist of a repair to the hardware, firmware, or software portions of the TOE, a modification of TOE guidance, or both. Corrective action that constitutes modifications to TOE guidance (e.g. details of procedural measures to be taken to obviate the security flaw) includes both those measures serving as only an interim solution (until the repair is issued) as well as those

serving as a permanent solution (where it is determined that the procedural measure is the best solution).

23 If the source of the security flaw is a documentation error, the corrective action consists of an update of the affected TOE guidance. If the corrective action is a procedural measure, this measure will include an update made to the affected TOE guidance to reflect these corrective procedures.

ALC_FLR.2.4C - *The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to the TOE users.*

ALC_FLR.2-5 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes a means of providing the TOE users with the necessary information on each security flaw.

24 The *necessary information* about each security flaw consists of its description (not necessarily at the same level of detail as that provided as part of work unit ALC_FLR.2-2), the prescribed corrective action, and any associated guidance on implementing the correction.

25 TOE users may be provided such information, correction, and documentation updates in any of several ways, such as their posting to a website, their being sent to TOE users, or arrangements made for the developer to install the correction. In cases where the means of providing this information requires action to be initiated by the TOE user, the evaluator examines any TOE guidance to ensure that it contains instructions for retrieving the information.

26 The only metric for assessing the adequacy of the method used for providing the information, corrections and guidance is that there be a reasonable expectation that TOE users can obtain or receive it. For example, consider the method of dissemination where the requisite data is posted to a website for one month, and the TOE users know that this will happen and when this will happen. This may not be especially reasonable or effective (as, say, a permanent posting to the website), yet it is feasible that the TOE user could obtain the necessary information. On the other hand, if the information were posted to the website for only one hour, yet TOE users had no way of knowing this or when it would be posted, it is infeasible that they would ever get the necessary information.

ALC_FLR.2.5C - *The flaw remediation procedures documentation shall describe a means by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.*

ALC_FLR.2-6 The evaluator shall examine the flaw remediation procedures to determine that they describe procedures for the developer to accept reports of security flaws or requests for corrections to such flaws.

27 The procedures ensure that TOE users have a means by which they can communicate with the TOE developer. By having a means of contact with the developer, the user can report security flaws, enquire about the status of security flaws, or request corrections to flaws. This means of contact may be part of a more general contact facility for reporting non-security related problems.

28 The use of these procedures is not restricted to TOE users; however, only the TOE users are actively supplied with the details of these procedures. Others who might have access to or familiarity with the TOE can use the same procedures to submit reports to the developer, who is then expected to process them. Any means of submitting reports to the developer, other than those identified by the developer, are beyond the scope of this work unit; reports generated by other means need not be addressed.

ALC_FLR.2.6C - *The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.*

ALC_FLR.2-7 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure every reported flaw is corrected.

29 The flaw remediation procedures cover not only those security flaws discovered and reported by developer personnel, but also those reported by TOE users. The procedures are sufficiently detailed so that they describe how it is ensured that each reported security flaw is corrected. The procedures contain reasonable steps that show progress leading to the eventual, inevitable resolution.

30 The procedures describe the process that is taken from the point at which the suspected security flaw is determined to be a security flaw to the point at which it is resolved.

ALC_FLR.2-8 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure that the TOE users are issued corrective actions for each security flaw.

31 The procedures describe the process that is taken from the point at which a security flaw is resolved to the point at which the corrective action is provided. The procedures for delivering corrective actions should be consistent with the security objectives; they need not necessarily be identical to the procedures used for delivering the TOE, as documented to meet ADO_DEL, if included in the assurance requirements. For example, if the hardware portion of a TOE were originally delivered by bonded courier, updates to hardware resulting from flaw remediation would likewise expected to be distributed by bonded courier. Updates unrelated to flaw remediation would follow the procedures set forth in the documentation meeting the ADO_DEL requirements.

ALC_FLR.2.7C - *The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.*

ALC_FLR.2-9 **The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would result in safeguards that the potential correction contains no adverse effects.**

32 Through analysis, testing, or a combination of the two, the developer may reduce the likelihood that adverse effects will be introduced when a security flaw is corrected. The evaluator assesses whether the procedures provide detail in how the necessary mix of analysis and testing actions is to be determined for a given correction.

33 The evaluator also determines that, for instances where the source of the security flaw is a documentation problem, the procedures include the means of safeguarding against the introduction of contradictions with other documentation.

ALC_FLR.2.8C - *The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.*

ALC_FLR.2-10 **The evaluator shall examine the flaw remediation guidance to determine that the application of these procedures would result in a means for the TOE user to provide reports of suspected security flaws or requests for corrections to such flaws.**

The guidance ensures that TOE users have a means by which they can communicate with the TOE developer. By having a means of contact with the developer, the user can report security flaws, enquire about the status of security flaws, or request corrections to flaws.

0.3 Evaluation of flaw remediation (ALC_FLR.3)

0.3.1 Objectives

34 The objective of this sub-activity is to determine whether the developer has established flaw remediation procedures that describe the tracking of security flaws, the identification of corrective actions, and the distribution of corrective action information to TOE users. Additionally, this sub-activity determines whether the developer's procedures provide for the corrections of security flaws, for the receipt of flaw reports from TOE users, for assurance that the corrections introduce no new security flaws, for the establishment of a point of contact for each TOE user, and for the timely issue of corrective actions to TOE users.

35 In order for the developer to be able to act appropriately upon security flaw reports from TOE users, TOE users need to understand how to submit security flaw reports to the developer, and developers need to know how to receive these reports. Flaw remediation guidance addressed to the TOE user ensures that TOE users are aware of how to communicate with the developer; flaw remediation procedures describe the developer's role is such communication.

0.3.2 Input

36 The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation;

b) flaw remediation guidance documentation.

0.3.3 Evaluator actions

37 This sub-activity comprises one CC Part 3 evaluator action element:

a) ALC_FLR.3.1E.

0.3.3.1 Action ALC_FLR.3.1E

ALC_FLR.3.1C - *The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.*

ALC_FLR.3-1 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes the procedures used to track all reported security flaws in each release of the TOE.

38 The procedures describe the actions that are taken by the developer from the time each suspected security flaw is reported to the time that it is resolved. This includes the flaw's entire timeframe, from initial detection through ascertaining the flaw is a security flaw, to resolution of the security flaw.

39 If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the ALC_FLR requirements) for the flaw remediation procedures to

track it further; only that there be an explanation of why the flaw is not security-relevant.

40 ALC_FLR.3.2C - *The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.*

ALC_FLR.3-2 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would produce a description of each security flaw in terms of its nature and effects.

41 The procedures identify the actions that are taken by the developer to describe the nature and effects of each security flaw in sufficient detail to be able to reproduce it. The description of the nature of a security flaw addresses whether it is an error in the documentation, a flaw in the design of the TSF, a flaw in the implementation of the TSF, etc. The description of the security flaw's effects identifies the portions of the TSF that are affected and how those portions are affected. For example, a security flaw in the implementation might be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password 'BACKDOOR'.

ALC_FLR.3-3 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would identify the status of finding a correction to each security flaw.

42 The flaw remediation procedures identify the different stages of security flaws. This differentiation includes at least: suspected security flaws that have been reported, suspected security flaws that have been confirmed to be security flaws, and security flaws whose solutions have been implemented. It is permissible that additional stages (e.g. flaws that have been reported but not yet investigated, flaws that are under investigation, security flaws for which a solution has been found but not yet implemented) be included.

ALC_FLR.3.3C - *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

ALC_FLR.3-4 The evaluator shall check the flaw remediation procedures to determine that the application of these procedures would identify the corrective action for each security flaw.

43 *Corrective action* may consist of a repair to the hardware, firmware, or software portions of the TOE, a modification of TOE guidance, or both. Corrective action that constitutes modifications to TOE guidance (e.g. details of procedural measures to be taken to obviate the security flaw) includes both those measures serving as only an interim solution (until the repair is issued) as well as those

serving as a permanent solution (where it is determined that the procedural measure is the best solution).

44 If the source of the security flaw is a documentation error, the corrective action consists of an update of the affected TOE guidance. If the corrective action is a procedural measure, this measure will include an update made to the affected TOE guidance to reflect these corrective procedures.

ALC_FLR.3.4C - *The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to the TOE users.*

ALC_FLR.3-5 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes a means of providing the TOE users with the necessary information on each security flaw.

45 The *necessary information* about each security flaw consists of its description (not necessarily at the same level of detail as that provided as part of work unit ALC_FLR.3-2), the prescribed corrective action, and any associated guidance on implementing the correction.

46 TOE users may be provided such information, correction, and documentation updates in any of several ways, such as their posting to a website, their being sent to TOE users, or arrangements made for the developer to install the correction. In cases where the means of providing this information requires action to be initiated by the TOE user, the evaluator examines any TOE guidance to ensure that it contains instructions for retrieving the information.

47 The only metric for assessing the adequacy of the method used for providing the information, corrections and guidance is that there be a reasonable expectation that TOE users can obtain or receive it. For example, consider the method of dissemination where the requisite data is posted to a website for one month, and the TOE users know that this will happen and when this will happen. This may not be especially reasonable or effective (as, say, a permanent posting to the website), yet it is feasible that the TOE user could obtain the necessary information. On the other hand, if the information were posted to the website for only one hour, yet TOE users had no way of knowing this or when it would be posted, it is infeasible that they would ever get the necessary information.

48 For TOE users who register with the developer (see work unit ALC_FLR.3-12), the passive availability of this information is not sufficient. Developers must actively send the information (or a notification of its availability) to registered TOE users.

ALC_FLR.3.5C - *The flaw remediation procedures documentation shall describe a means by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.*

ALC_FLR.3-6 The evaluator shall examine the flaw remediation procedures to determine that they describe procedures for the developer to accept reports of security flaws or requests for corrections to such flaws.

49 The procedures ensure that TOE users have a means by which they can communicate with the TOE developer. By having a means of contact with the developer, the user can report security flaws, enquire about the status of security flaws, or request corrections to flaws. This means of contact may be part of a more general contact facility for reporting non-security related problems.

50 The use of these procedures is not restricted to TOE users; however, only the TOE users are actively supplied with the details of these procedures. Others who might have access to or familiarity with the TOE can use the same procedures to submit reports to the developer, who is then expected to process them. Any means of submitting reports to the developer, other than those identified by the developer, are beyond the scope of this work unit; reports generated by other means need not be addressed.

ALC_FLR.3.6C - *The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.*

ALC_FLR.3-7 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure that every reported flaw is corrected.

51 The flaw remediation procedures cover not only those security flaws discovered and reported by developer personnel, but also those reported by TOE users. The procedures are sufficiently detailed so that they describe how it is ensured that each reported security flaw is corrected. The procedures contain reasonable steps that show progress leading to the eventual, inevitable resolution.

52 The procedures describe the process that is taken from the point at which the suspected security flaw is determined to be a security flaw to the point at which it is resolved.

ALC_FLR.3-8 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure that the TOE users are issued corrective actions for each security flaw.

53 The procedures describe the process that is taken from the point at which a security flaw is resolved to the point at which the corrective action is provided. The procedures for delivering corrective actions should be consistent with the security objectives; they need not necessarily be identical to the procedures used for

delivering the TOE, as documented to meet ADO_DEL, if included in the assurance requirements. For example, if the hardware portion of a TOE were originally delivered by bonded courier, updates to hardware resulting from flaw remediation would likewise expected to be distributed by bonded courier. Updates unrelated to flaw remediation would follow the procedures set forth in the documentation meeting the ADO_DEL requirements.

ALC_FLR.3.7C - *The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.*

ALC_FLR.3-9 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would result in safeguards that the potential correction contains no adverse effects.

54 Through analysis, testing, or a combination of the two, the developer may reduce the likelihood that adverse effects will be introduced when a security flaw is corrected. The evaluator assesses whether the procedures provide detail in how the necessary mix of analysis and testing actions is to be determined for a given correction.

55 The evaluator also determines that, for instances where the source of the security flaw is a documentation problem, the procedures include the means of safeguarding against the introduction of contradictions with other documentation.

ALC_FLR.3.8C - *The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.*

ALC_FLR.3-10 The evaluator shall examine the flaw remediation guidance to determine that the application of these procedures would result in a means for the TOE user to provide reports of suspected security flaws or requests for corrections to such flaws.

The guidance ensures that TOE users have a means by which they can communicate with the TOE developer. By having a means of contact with the developer, the user can report security flaws, enquire about the status of security flaws, or request corrections to flaws.

ALC_FLR.3.9C - *The flaw remediation procedures shall include a procedure requiring timely responses for the automatic distribution of security flaw reports and the associated corrections to registered users who might be affected by the security flaw.*

ALC_FLR.3-11 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would result in a timely means of providing the registered TOE users who might be affected with reports about, and associated corrections to, each security flaw.

56 The issue of timeliness applies to the issuance of both security flaw reports and the associated corrections. However, these need not be issued at the same time. It is recognised that flaw reports should be generated and issued as soon as an interim solution is found, even if that solution is as drastic as ‘Turn off the TOE’. Likewise, when a more permanent (and less drastic) solution is found, it should be issued without undue delay.

57 It is unnecessary to restrict the recipients of the reports and associated corrections to only those TOE users who might be affected by the security flaw; it is permissible that all TOE users be given such reports and corrections for all security flaws, provided such is done in a timely manner.

ALC_FLR.3-12 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would result in automatic distribution of the reports and associated corrections to the registered TOE users who might be affected.

58 *Automatic distribution* does not mean that human interaction with the distribution method is not permitted. In fact, the distribution method could consist entirely of manual procedures, perhaps through a closely monitored procedure with prescribed escalation upon the lack of issue of reports or corrections.

59 It is unnecessary to restrict the recipients of the reports and associated corrections to only those TOE users who might be affected by the security flaw; it is permissible that all TOE users be given such reports and corrections for all security flaws, provided such is done automatically.

ALC_FLR.3.10C - *The flaw remediation guidance shall describe a means by which TOE users may register with the developer, to be eligible to receive security flaw reports and corrections.*

ALC_FLR.3-13 The evaluator shall examine the flaw remediation guidance to determine that it describes a means of enabling the TOE users to register with the developer.

60 *Enabling the TOE users to register with the developer* simply means having a way for each TOE user to provide the developer with a point of contact; this point of contact is to be used to provide the TOE user with information related to security flaws that might affect that TOE user, along with any corrections to the security flaw. Registering the TOE user may be accomplished as part of the standard procedures that TOE users undergo to identify themselves to the developer, for the purposes of registering a software licence, or for obtaining update and other useful information.

61 There need not be one registered TOE user per installation of the TOE; it would be sufficient if there were one registered TOE user for an organisation. For example,

a corporate TOE user might have a centralised acquisition office for all of its sites. In this case, the acquisition office would be a sufficient point of contact for all of that TOE user's sites, so that all of the TOE user's installations of the TOE have a registered point of contact.

62 In either case, it must be possible to associate each TOE that is delivered with an organisation in order to ensure that there is a registered user for each TOE. For organisations that have many different addresses, this assures that there will be no user who is erroneously presumed to be covered by a registered TOE user.

63 It should be noted that TOE users need not register; they must only be provided with a means of doing so. However, users who choose to register must be directly sent the information (or a notification of its availability).

ALC_FLR.3.11C - *The flaw remediation guidance shall identify the specific points of contact for all reports and enquiries about security issues involving the TOE.*

ALC_FLR.3-14 The evaluator shall examine the flaw remediation guidance to determine that it identifies specific points of contact for reports and enquiries about security issues involving the TOE.

64 The guidance includes a means whereby registered TOE users can interact with the developer to report discovered security flaws in the TOE or to make enquiries regarding discovered security flaws in the TOE.