Common Methodology for Information Technology Security Evaluation

---

Evaluation methodology

November 2022

CEM:2022
Revision 1

CCMB-2022-11-006

Contents

Foreword

This version of the Common Methodology for Information Technology Security Evaluation (CEM:2022) is the first major revision since being published as CEM v3.1 Revision 5 in 2017.

Historically, the CC standard along with the Common Evaluation Methodology (CEM) was developed and maintained by the participating nations of the Agreement on the Recognition of Common Criteria Certificates in the field of IT Security (CCRA) and subsequently published as standards maintained by ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission). CC:2022 and CEM:2022, however, were developed first as ISO/IEC standards and subsequently published by the CCRA as the new version of the CC and CEM. The ISO version of the CC:2022 is published in five parts as ISO/IEC 15408-1:2022 through 15408-5:2022 and the ISO version of the CEM:2022 is published in one part as ISO/IEC 18045:2022.

The main changes in CEM:2022 are as follows:

— the exact conformance type has been introduced;

— low assurance PPs have been removed and direct rationale PPs have been introduced;

— updated evaluation methodology for PP-modules and PP-configurations for modular evaluations has been introduced;

— multi-assurance evaluation has been introduced.

— composition of assurance has been introduced.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

Legal notice

The governmental organizations listed below contributed to the development of this version of the Common Methodology for Information Technology Security Evaluation. As the joint holders, together with ISO/IEC, of the copyright in the Common Methodology for Information Technology Security Evaluation, version 2022 (called "CEM:2022"), they hereby grant a non-exclusive permission to ISO/IEC to reproduce CEM:2022 in the revised editions of ISO/IEC 18045 and its derivatives, including their national adoptions. However, these governmental organizations retain the right to use, copy, distribute, translate or modify CEM:2022 as they see fit. ISO/IEC has in return granted permission to the aforementioned organizations to license the resulting CEM:2022 under any licence they may see appropriate. The aforementioned governmental organizations have always been supportive of the text being reused by any users of the document, including modifications and reuse of part of the document, and will continue to follow this policy.

Australia	The Australian Signals Directorate
Canada	Communications Security Establishment
France	Agence Nationale de la Sécurité des Systèmes d'Information
Germany	Bundesamt für Sicherheit in der Informationstechnik
Japan	Information-technology Promotion Agency
Netherlands	Netherlands National Communications Security Agency
New Zealand	Government Communications Security Bureau
Republic of Korea	National Security Research Institute
Spain	Ministerio de Asuntos Económicos y Transformación Digital and Centro Criptológico Nacional
Sweden	FMV, Swedish Defence Materiel Administration
United Kingdom	National Cyber Security Centre
United States	The National Security Agency and the National Institute of Standards and Technology

Introduction

The target audience for this document is primarily evaluators applying the CC and certifiers confirming evaluator actions. Evaluation sponsors, developers, protection profile (PP), PP-Module, PP-Configuration, and security target (ST) authors, and other parties interested in IT security, can be a secondary audience.

This document cannot answer all questions concerning IT security evaluation and further interpretations may be needed. Individual schemes determine how to handle such interpretations, although these can be subject to mutual recognition agreements. A list of methodology-related activities that can be handled by individual schemes can be found in Annex A.

This document is intended to be used in conjunction with the CC.

NOTE This document uses bold and italic type in some cases to distinguish terms from the rest of the text. The relationship between components within a family is highlighted using a bolding convention. This convention calls for the use of bold type for all new requirements. For hierarchical components, requirements are presented in bold type when they are enhanced or modified beyond the requirements of the previous component. In addition, any new or enhanced permitted operations beyond the previous component are also highlighted using bold type.

The use of italics indicates text that has a precise meaning. For security assurance requirements the convention is for special verbs relating to evaluation.

Common Methodology for Information Technology Security Evaluation — Evaluation methodology

1 Scope

This document defines 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 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Common Criteria for Information Technology Security Evaluation, CC:2022, revision 1, November 2022 — Part 1: Introduction and general model

Common Criteria for Information Technology Security Evaluation, CC:2022, revision 1, November 2022 — Part 2: Security functional components

Common Criteria for Information Technology Security Evaluation, CC:2022, revision 1, November 2022 — Part 3: Security assurance components

Common Criteria for Information Technology Security Evaluation, CC:2022, revision 1, November 2022 — Part 4: Framework for the specification of evaluation methods and activities

Common Criteria for Information Technology Security Evaluation, CC:2022, revision 1, November 2022 — Part 5: Pre-defined packages of security requirements

ISO/IEC IEEE 24765, *Systems and software engineering — Vocabulary*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in CC Part 1, CC Part 2, CC Part 3, CC Part 4, CC Part 5, ISO/IEC IEEE 24765, and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- IEC Electropedia: available at https://www.electropedia.org/
- ISO Online browsing platform: available at https://www.iso.org/obp/

3.1
check, verb
<evaluation> generate a verdict by a simple comparison

Note 1 to entry: Evaluator expertise is not required. The statement that uses this verb describes what is mapped.

3.2
confirm, verb
<evaluation> declare that something has been reviewed in detail with an independent determination of sufficiency

Note 1 to entry: The level of rigour required depends on the nature of the subject matter.

3.3
demonstrate, verb
<evaluation> provide a conclusion gained by an analysis which is less rigorous than a "proof"

3.4
describe, verb
<evaluation> provide specific details of an entity

3.5
determine, verb
<evaluation> affirm a particular conclusion based on independent analysis with the objective of reaching a particular conclusion

Note 1 to entry: The usage of this term implies a truly independent analysis, usually in the absence of any previous analysis having been performed. Compare with the terms "*confirm*" (3.2) or "verify" which imply that an analysis has already been performed which needs to be reviewed.

3.6
encountered potential vulnerability
potential weakness in the target of evaluation (TOE) identified by the evaluator while performing evaluation activities that can be used to violate the security functional requirements (SFRs)

3.7
ensure, verb
<evaluation> guarantee a strong causal relationship between an action and its consequences

Note 1 to entry: When "ensure" is preceded by the word "help" it indicates that the consequence is not fully certain, on the basis of that action alone.

3.8
evaluation evidence
item used as a basis for establishing the verdict of an evaluation activity

3.9
examine, verb
<evaluation> generate a verdict by analysis using evaluator expertise

Note 1 to entry: The statement that uses this verb identifies what is analysed and the properties for which it is analysed.

3.10
exhaustive, adj.
<evaluation> characteristic of a methodical approach taken to perform an analysis or activity according to an unambiguous plan

Note 1 to entry: This term is used in respective parts of the CC with respect to conducting an analysis or other activity. It is related to "systematic" but is considerably stronger, in that it indicates not only that a methodical approach has been taken to perform the analysis or activity according to an unambiguous plan, but that the plan that was followed is sufficient to *ensure* (3.7) that all possible avenues have been exercised.

3.11
explain, verb
<evaluation> give argument accounting for the reason for taking a course of action

Note 1 to entry: This term differs from both "*describe*" (3.4) and "*demonstrate*" (3.3). It is intended to answer the question "Why?" without actually attempting to argue that the course of action that was taken was necessarily optimal.

3.12
justify, verb
<evaluation> provide a rationale providing sufficient reason

Note 1 to entry: The term 'justify' is more rigorous than '*demonstrate*' (3.3). This term requires significant rigour in terms of very carefully and thoroughly *explaining* (3.11) every step of a logical analysis leading to a conclusion.

3.13
monitoring attack
generic category of attack methods that includes passive analysis techniques aiming at disclosure of sensitive internal data of the target of evaluation (TOE) by operating the TOE in the way that corresponds to the guidance documents

3.14
observation report
OR
report written by the evaluator requesting a clarification or identifying a problem during the evaluation

3.15
oversight verdict
statement issued by an evaluation authority confirming or rejecting an overall verdict based on the results of evaluation oversight activities

3.16
prove, verb
<evaluation> show correspondence by formal analysis in its mathematical sense

Note 1 to entry: It is completely rigorous in all ways. Typically, the term "prove" is used when there is a desire to show correspondence between two target of evaluation (TOE) security functionality (TSF) representations at a high level of rigour.

3.17
record, verb
<evaluation> 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

3.18
report, verb
<evaluation> include evaluation results and supporting material in the evaluation technical report, an *observation report* (3.14) or an evaluation authority report (report of the evaluation authority)

3.19
specify, verb
<evaluation> provide specific details about an entity in a rigorous and precise manner

3.20
trace, verb
<evaluation> establish a relation between two sets of entities, which shows which entities in the first set correspond to which entities in the second

3.21
verdict
statement issued by an evaluator with respect to evaluator action element, assurance component, or class

3.22
verify, verb
<evaluation> rigorously review in detail with an independent determination of sufficiency

Note 1 to entry: Also see "*confirm*" (3.2). This term has more rigorous connotations. The term "verify" is used in the context of evaluator actions where an independent effort is required of the evaluator.

3.23
window of opportunity
period of time that an attacker has access to the target of evaluation (TOE)

3.24
work unit
most granular level of evaluation work

4 Abbreviated terms

OR observation report

5 Terminology

Unlike in the CC, where each element maintains the last digit of its identifying symbol for all components within the family, this document can 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 can change although the work unit remains unchanged.

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

6 Verb usage

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.

Guidance text accompanying work units and sub-tasks gives further explanation on how to apply CC words in an evaluation. The verb usage is in accordance with ISO definitions for these verbs. The auxiliary verb *should* is used to indicate a recommendation. *May* indicates a permission and *can* indicates a possibility.

The verbs *check*, *examine*, *report* and *record* are used with a precise meaning within this part of this document and Clause 3 should be referenced for their definitions.

7 General evaluation guidance

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 A. 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.

8 Relationship between the CC and CEM structures

There are direct relationships between the CC structure (i.e. class, family, component and element) and the structure of this document. Figure 1 illustrates the correspondence between the CC constructs of class, family and evaluator action elements and evaluation methodology activities, sub-activities and actions. However, several evaluation methodology work units can result from the requirements noted in the CC developer action and content and presentation elements.

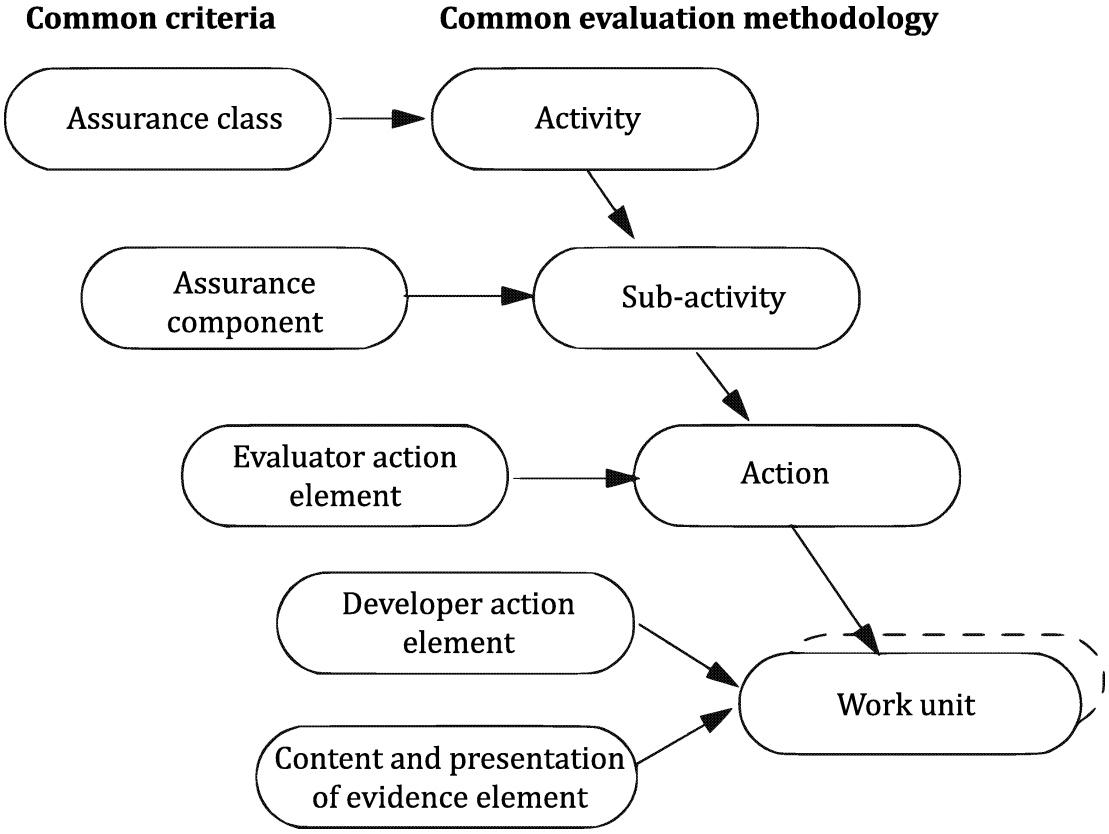

Figure 1 — Mapping of the CC and CEM structures

9 Evaluation process and related tasks

This clause provides an overview of the evaluation process and defines the tasks an evaluator is intended to perform when conducting an evaluation.

Each evaluation, whether of a PP, PP-Configuration, or TOE (including ST), follows the same process, and has four evaluator tasks in common: the input task, the output task, the evaluation sub-activities, and the demonstration of the technical competence to the evaluation authority task.

The input task and the output tasks, which are related to management of evaluation evidence and to report generation, are entirely described in this clause. Each task has associated sub-tasks that apply to all CC evaluations (evaluation of a PP, PP-Configuration, or a TOE).

The evaluation sub-activities are only introduced in this clause, and fully described in the following chapters.

In contrast to the evaluation sub-activities, input and output tasks have no verdicts associated with them as they do not map to CC evaluator action elements; they are performed in order to ensure conformance with the universal principles and to be in conformance with this document.

The demonstration of the technical competence to the evaluation authority task can be fulfilled by the evaluation authority analysis of the output tasks results, or may include the demonstration by the evaluators of their understanding of the inputs for the evaluation sub-activities. This task has no associated evaluator verdict, but has an evaluator authority verdict. The detailed criteria to pass this task are left to the discretion of the evaluation authority, as noted in A.6.

Evaluation activities defined in conformance with CC Part 4 can be used in place of work units defined within this document provided that this is made clear within the evaluation and certification reports.

9.1 Evaluation process overview

9.1.1 Objectives

This subclause presents the general model of the methodology and identifies:

a) roles and responsibilities of the parties involved in the evaluation process;

b) the general evaluation model.

9.1.2 Responsibilities of the roles

The general model defines the following roles: sponsor, developer, evaluator and evaluation authority.

The sponsor is responsible for requesting and supporting an evaluation. This means that the sponsor establishes the different agreements for the evaluation (e.g. commissioning the evaluation). Moreover, the sponsor is responsible for ensuring that the evaluator is provided with the evaluation evidence.

The developer produces the TOE and is responsible for providing the evidence required for the evaluation (e.g. training, design information), on behalf of the sponsor.

The evaluator performs the evaluation tasks required in the context of an evaluation: the evaluator receives the evaluation evidence from the developer on behalf of the sponsor or directly from the sponsor, performs the evaluation sub-activities and provides the results of the evaluation assessment to the evaluation authority.

The evaluation authority establishes and maintains the scheme, monitors the evaluation conducted by the evaluator, and issues certification/validation reports as well as certificates based on the evaluation results provided by the evaluator.

9.1.3 Relationship of roles

To prevent undue influence from improperly affecting an evaluation, some separation of roles is required. This implies that the roles described above are fulfilled by different entities, except that the roles of developer and sponsor can be satisfied by a single entity.

Moreover, some evaluations (e.g. EAL1 evaluation) may not require the developer to be involved in the project. In this case, it is the sponsor who provides the TOE to the evaluator and who generates the evaluation evidence.

9.1.4 General evaluation model

The evaluation process consists of the evaluator performing the evaluation input task, the evaluation output task and the evaluation sub-activities. Figure 2 provides an overview of the relationship between these tasks and sub-activities.

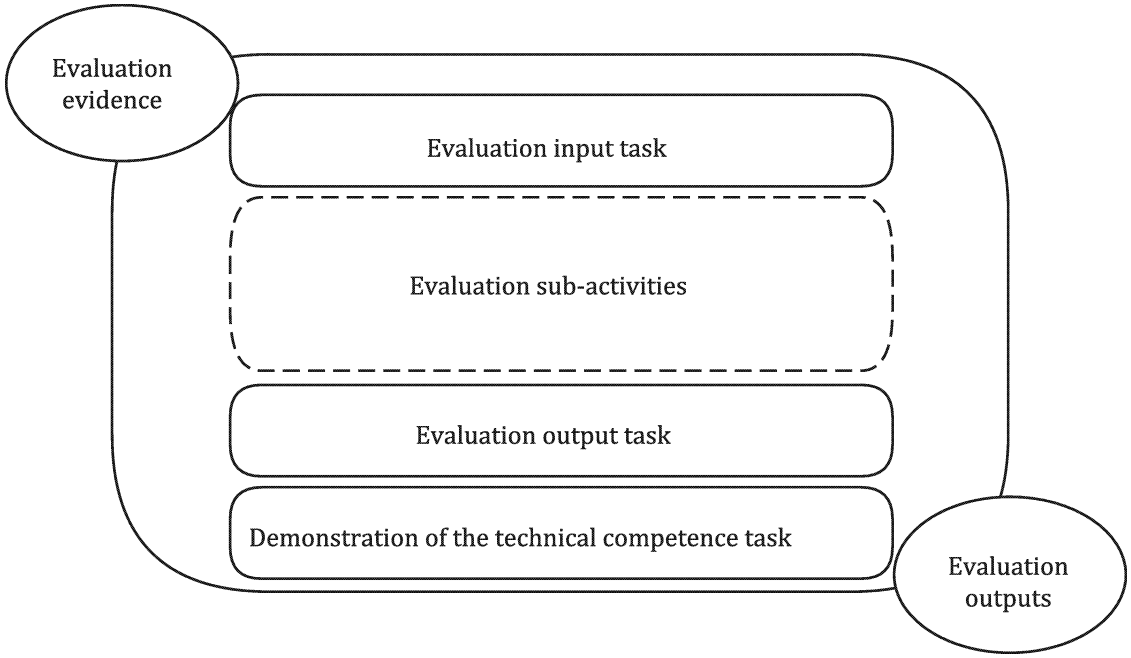

Figure 2 — Generic evaluation model

The evaluation process may be preceded by a preparation phase where initial contact is made between the sponsor and the evaluator. The work that is performed and the involvement of the different roles during this phase may vary. It is typically during this step that the evaluator performs a feasibility analysis to assess the likelihood of a successful evaluation.

9.1.5 Evaluator verdicts

The evaluator assigns verdicts to the requirements of the CC and not to those of this document. 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 evaluation methodology action and its constituent work units. Where required by a Security Target, the evaluator may follow Evaluation Activities derived from work units in this document (for example using the framework in CC Part 4). Where such derived

Evaluation Methods and Evaluation Activities are required by an ST then these shall be used by the evaluator to assign verdicts to the relevant evaluator action element(s). Finally, an evaluation result is assigned, as described in CC Part 1, Clause 13, Evaluation and evaluation results.

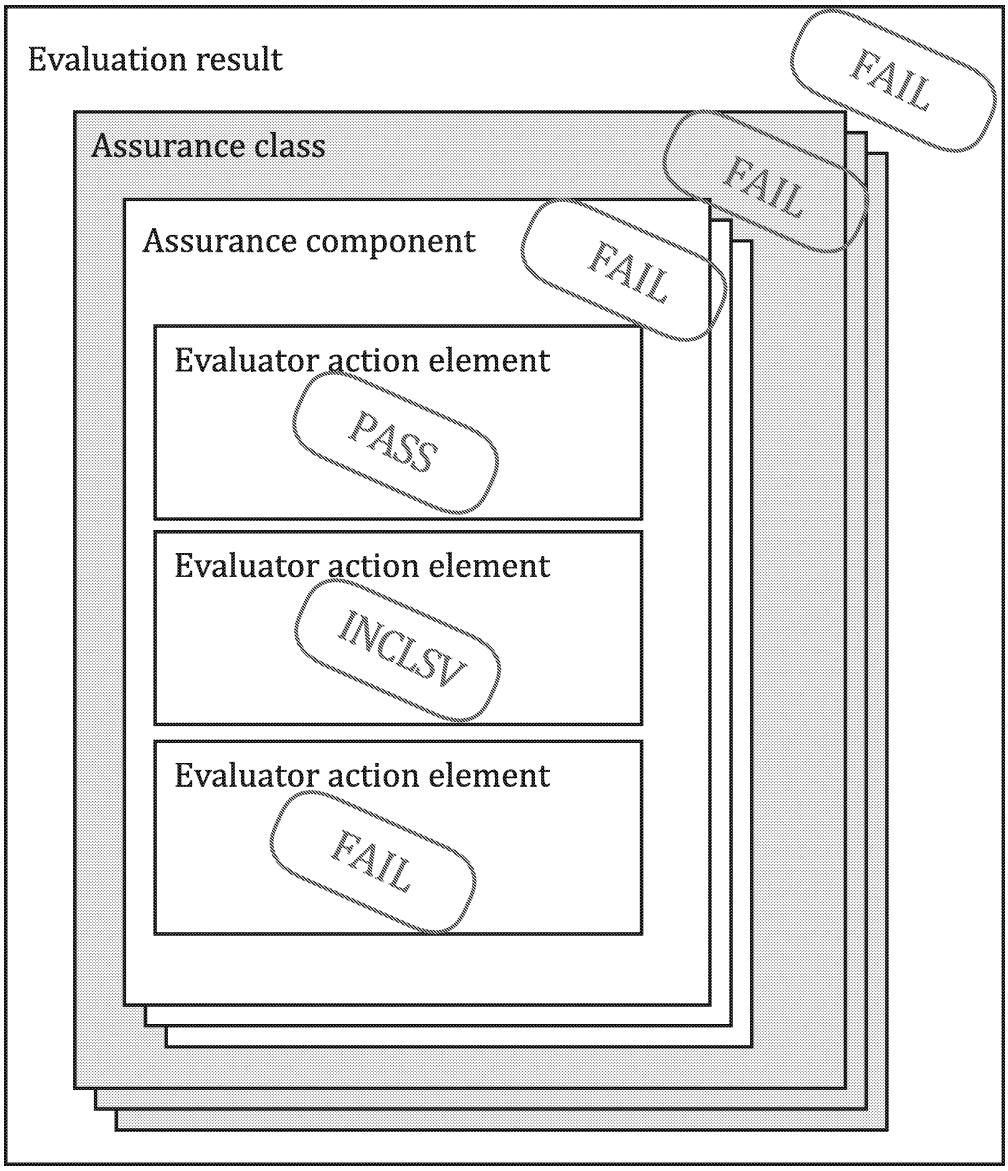

Figure 3 — Example of the verdict assignment rule

This document recognises three mutually exclusive verdict states:

a) conditions for a pass verdict are defined as an evaluator completion of CC evaluator action element and determination that the requirements for the PP, PP-Configuration, ST or TOE under evaluation are met. The conditions for passing the element are defined as:

   1) the constituent work units of the related evaluation methodology action, and;

   2) all evaluation evidence required for performing these work units is coherent, that is it can be fully and completely understood by the evaluator, and;

   3) all evaluation evidence required for performing these work units does not have any obvious internal inconsistencies or inconsistencies with other evaluation evidence.

Note that obvious means here that the evaluator discovers this inconsistency while performing the work units: the evaluator should not undertake a full consistency analysis across the entire evaluation evidence every time a work unit is performed.

b) conditions for a *fail* verdict are defined as an evaluator completion of CC evaluator action element and determination that the requirements for the PP, PP-Configuration, ST, or TOE under evaluation are not met, or that the evidence is incoherent, or an obvious inconsistency in the evaluation evidence has been found;

c) all verdicts are initially *inconclusive* and remain so until either a *pass* or *fail* verdict is assigned.

The overall verdict is *pass* if and only if all the constituent verdicts are also *pass*. In the example illustrated in Figure 3, 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*.

9.2 Evaluation input task

9.2.1 Objectives

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.

9.2.2 Application notes

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, all evaluation evidence pertaining to all parts of the TOE is to be 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 parts of the TOE. For example, if design is required, then the TOE design (ADV_TDS) requirements will apply to all subsystems that are part of the TSF. In addition, assurance requirements that call for procedures to be in place, e.g. CM capabilities (ALC_CMC) and Delivery (ALC_DEL) will also apply to the entire TOE (including any part produced by another developer).

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 subclauses of interest) to help the evaluator to find easily the required evidence.

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.

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.

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).

9.2.3 Management of evaluation evidence sub-task

9.2.3.1 Configuration control

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

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.

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

9.2.3.2 Disposal

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.

9.2.3.3 Confidentiality

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.

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

9.3 Evaluation sub-activities

The evaluation sub-activities vary depending on whether it is a PP, PP-Configuration, or a TOE evaluation. Moreover, in the case of a TOE evaluation, the sub-activities depend upon the selected assurance requirements.

9.4 Evaluation output task

9.4.1 Objectives

The objective of this subclause 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. This document does not preclude the addition of information into these reports as this International Standard specifies only the minimum information content.

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. The consistency of ETR and OR among different evaluations is the responsibility of the evaluation authority.

The evaluator performs the two following sub-tasks in order to meet the requirements of this document for the information content of reports:

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

b) write ETR sub-task.

9.4.2 Management of evaluation outputs

The evaluator delivers the ETR to the evaluation authority, as well as any ORs as they become available. Requirements for controls on handling the ETR and ORs are established by the scheme which may include delivery to the sponsor or developer. The ETR and ORs may include sensitive or proprietary information and may need to be sanitised before they are given to the sponsor.

9.4.3 Application notes

In this edition of this document, the requirements for the provision of evaluator evidence to support re-evaluation and re-use have not been explicitly stated. 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.

9.4.4 Write OR sub-task

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

In the case of a fail verdict, the evaluator ***shall provide*** an OR to reflect the evaluation result. Otherwise, the evaluator may use ORs as one way of expressing clarification needs.

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.

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 evaluation authorities and sponsors).

9.4.5 Write ETR sub-task

9.4.5.1 Objectives

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

This document 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.

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

The ETR supports the evaluation authority to confirm that the evaluation was done to the required standard, but it is anticipated that the documented results may not provide all of the necessary information, so additional information specifically requested by the scheme may be necessary. This aspect is outside the scope of this document.

9.4.5.2 ETR for a PP Evaluation

9.4.5.2.1 General

This subclause describes the minimum content of the ETR for a PP evaluation. The contents of the ETR are portrayed in Figure 4; this figure may be used as a guide when constructing the structural outline of the ETR document.

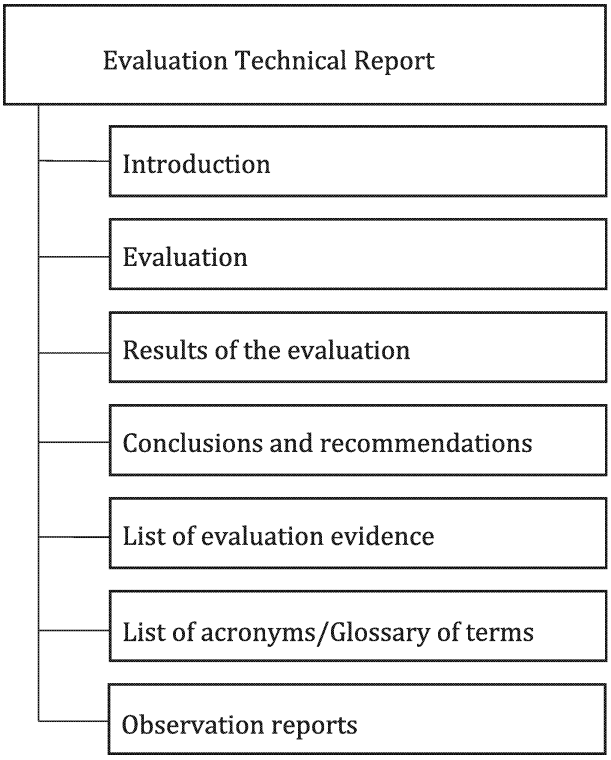

Figure 4 — ETR information content for a PP evaluation

9.4.5.2.2 General

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

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

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

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

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

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

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

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

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

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

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

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

9.4.5.2.3 Evaluation

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

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

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.

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

9.4.5.2.4 Results of the evaluation

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 evaluation methodology action and its constituent work units.

The rationale justifies the verdict using the CC, this document, 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 an evaluation methodology work unit.

9.4.5.2.5 Conclusions and recommendations

The evaluator ***shall report*** the conclusions of the evaluation, in particular the overall verdict as defined in CC Part 1, Clause 12, Evaluation results, and determined by application of the verdict assignment described in 9.1.5.

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

9.4.5.2.6 List of evaluation evidence

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

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

b) the title;

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

9.4.5.2.7 List of acronyms/Glossary of terms

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

Glossary definitions already defined by the CC (all parts) or by this document need not be repeated in the ETR.

9.4.5.2.8 Observation reports

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

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

9.4.5.3 ETR for a PP-Configuration Evaluation

9.4.5.3.1 General

This subclause describes the minimum content of the ETR for a PP-Configuration evaluation. The contents of the ETR are portrayed in Figure 5; this figure may be used as a guide when constructing the structural outline of the ETR document.

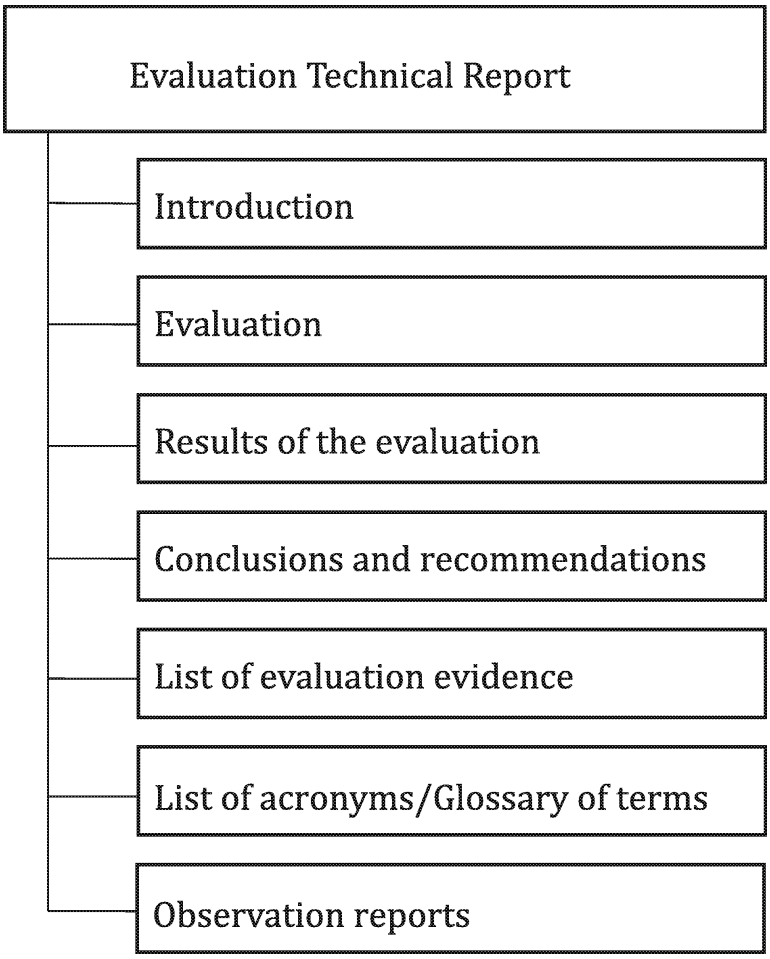

Figure 5 — ETR information content for a PP-Configuration evaluation

9.4.5.3.2 General

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

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

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

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

The evaluator ***shall report*** PP-Configuration configuration control identifiers.

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

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

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

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

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

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

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

9.4.5.3.3 PP-Configuration overview

The evaluator ***shall report*** all PP-Configuration components used in the PP-configuration.

PP-Configuration components include at least one PP, as well as all other PPs and PP-Modules used in the PP-Configuration. Each PP-Module will have one of its PP-Module Bases included in the PP-Configuration, therefore all elements of the relevant PP-Module Base are included as PP-Configuration components. See CC Part 1, Annex C for more detail on the identification of the PP-Module Base for each PP-Module.

The ETR **shall report** the reference of the PP-Configuration components.

A PP-Configuration component reference contains information that uniquely identifies the specific component (e.g. title, date, and version number).

9.4.5.3.4 Evaluation

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

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

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.

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

9.4.5.3.5 Results of the evaluation

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

The rationale justifies the verdict using the CC, this document, 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 an evaluation methodology work unit.

9.4.5.3.6 Conclusions and recommendations

The evaluator ***shall report*** the conclusions of the evaluation, in particular the overall verdict as defined in CC Part 1, Clause 12, Evaluation results, and determined by application of the verdict assignment described in 9.1.5 .

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

9.4.5.3.7 List of evaluation evidence

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

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

b) the title;

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

9.4.5.3.8 List of acronyms/Glossary of terms

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

Glossary definitions already defined by the CC or by this document need not be repeated in the ETR.

9.4.5.3.9 Observation reports

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

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

9.4.5.4 ETR for a TOE Evaluation

9.4.5.4.1 General

This subclause describes the minimum content of the ETR for a TOE evaluation. The contents of the ETR are portrayed in Figure 6; this figure may be used as a guide when constructing the structural outline of the ETR document.

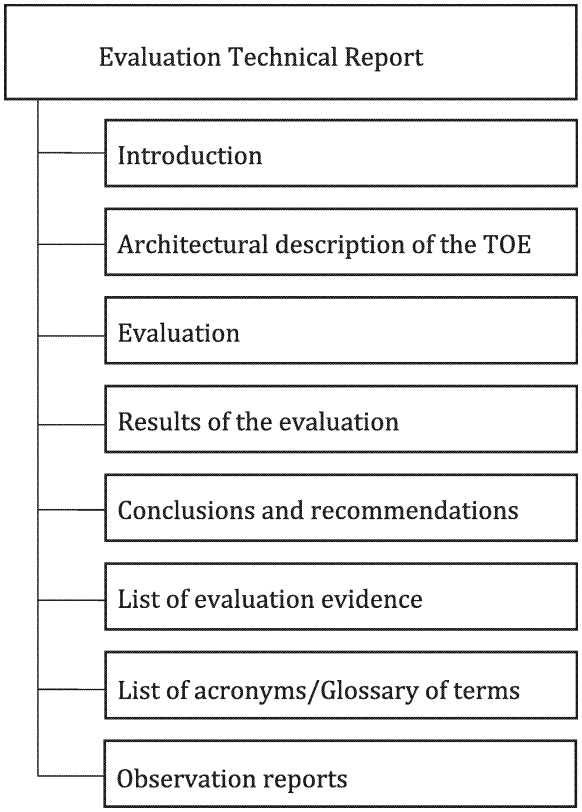

Figure 6 — ETR information content for a TOE evaluation

9.4.5.4.2 General

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

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

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

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

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

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

If the ST claims that the TOE conforms to the requirements of one or more PPs, or to a PP-Configuration, the ETR shall report the reference of the corresponding PPs/PP-Configuration.

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

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

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

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

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

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

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

9.4.5.4.3 Architectural description of the TOE

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 TOE design (ADV_TDS), where applicable.

The intent of this subclause is to characterise the degree of architectural separation of the major components. If there is no TOE design (ADV_TDS) requirement in the ST, this is not applicable and is considered to be satisfied.

9.4.5.4.4 Evaluation

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

The evaluator may reference the evaluation criteria, methodology and interpretations used to evaluate the TOE or the devices used to perform the tests. This shall include identification of any derived Evaluation Methods and Evaluation Activities used, and a confirmation that these are as required by the Security Target (cf. ASE_CCL).

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.

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

9.4.5.4.5 Results of the evaluation

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

a) the title of the activity considered;

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

The rationale justifies the verdict using the CC, this document, any derived Evaluation Methods and Evaluation Activities required by the ST, 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 an evaluation methodology work unit.

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

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

9.4.5.4.6 Conclusions and recommendations

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, Evaluation and evaluation results, and determined by application of the verdict assignment described in 9.1.5.

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

9.4.5.4.7 List of evaluation evidence

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

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

b) the title;

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

9.4.5.4.8 List of acronyms/Glossary of terms

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

Glossary definitions already defined by the CC or by this document need not be repeated in the ETR.

9.4.5.4.9 Observation reports

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

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

10 Class APE: Protection Profile evaluation

10.1 General

This clause 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 requirements) that is claimed in the PP. The evaluation methodology in this clause is based on the requirements on the PP as specified in CC Part 3 class APE.

This clause should be used in conjunction with CC Part 1, Annexes B and D, as these Annexes clarify the concepts here and provide many examples.

10.2 Re-using the evaluation results of certified PPs

While evaluating a PP that is based on one or more certified PPs, it may be possible to re-use the fact that these PPs were certified. The potential for re-use of the result of a certified PP is greater if the PP under evaluation does not add threats, OSPs, security objectives and/or security requirements to those of the PP that conformance is being claimed to. If the PP under evaluation contains much more than the certified PP, re-use may not be useful at all.

The evaluator is allowed to re-use the PP evaluation results by doing certain analyses only partially or not at all if these analyses or parts thereof were already done as part of the PP evaluation. While doing this, the evaluator should assume that the analyses in the PP were performed correctly.

An example would be where the PP that conformance is being claimed to contains a set of security requirements, and these were determined to be internally consistent during its evaluation. If the PP under evaluation uses the exact same requirements, the consistency analysis does not have to be repeated during the PP evaluation. If the PP under evaluation adds one or more requirements, or performs operations on these requirements, the analysis will have to be repeated. However, it may be possible to save work in this consistency analysis by using the fact that the original requirements are internally consistent. If the original requirements are internally consistent, the evaluator only has to determine that:

a) the set of all new and/or changed requirements is internally consistent, and

b) the set of all new and/or changed requirements is consistent with the original requirements.

The evaluator notes in the ETR each case where analyses are not done or only partially done for this reason.

10.3 PP introduction (APE_INT)

10.3.1 Evaluation of sub-activity (APE_INT.1)

10.3.1.1 Objectives

The objective of this sub-activity is to determine whether the PP is correctly identified, and whether the PP reference and TOE overview are consistent with each other.

10.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP.

10.3.1.3 Action APE_INT.1.1E

10.3.1.3.1 General

CC Part 3 APE_INT.1.1C: *The PP introduction shall contain a PP reference and a TOE overview.*

10.3.1.3.2 Work unit APE_INT.1-1

The evaluator ***shall check*** that the PP introduction contains a PP reference and a TOE overview.

CC Part 3 APE_INT.1.2C: *The PP reference shall uniquely identify the PP.*

10.3.1.3.3 Work unit APE_INT.1-2

The evaluator ***shall examine*** the PP reference to determine that it uniquely identifies the PP.

The evaluator determines that the PP reference identifies the PP itself, so that it may be easily distinguished from other PPs, and that it also uniquely identifies each version of the PP, e.g. by including a version number and/or a date of publication.

The PP should have some referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates).

CC Part 3 APE_INT.1.3C: *The TOE overview shall summarize the usage and major security features of the TOE.*

10.3.1.3.4 Work unit APE_INT.1-3

The evaluator ***shall examine*** the TOE overview to determine that it describes the usage and major security features of the TOE.

The TOE overview should briefly (i.e. several paragraphs) describe the usage and major security features expected of the TOE. The TOE overview should enable consumers and potential TOE developers to quickly determine whether the PP is of interest to them.

The evaluator determines that the overview is clear enough for TOE developers and consumers, and sufficient to give them a general understanding of the intended usage and major security features of the TOE.

CC Part 3 APE_INT.1.4C: *The TOE overview shall identify the TOE type.*

10.3.1.3.5 Work unit APE_INT.1-4

The evaluator ***shall check*** that the TOE overview identifies the TOE type.

CC Part 3 APE_INT.1.5C: *The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.*

10.3.1.3.6 Work unit APE_INT.1-5

The evaluator ***shall examine*** the TOE overview to determine that it identifies any non-TOE hardware/software/firmware available to the TOE.

While some TOEs may run stand-alone, other TOEs (notably software TOEs) need additional hardware, software or firmware to operate. In this subclause of the PP, the PP author lists all hardware, software, and/or firmware that will be available for the TOE to run on.

This identification should be detailed enough for potential consumers and TOE developers to determine whether their TOE may operate with the listed hardware, software and firmware.

10.4 Conformance claims (APE_CCL)

10.4.1 Evaluation of sub-activity (APE_CCL.1)

10.4.1.1 Objectives

The objective of this sub-activity is to determine the validity of various conformance claims. These describe how the PP conforms to the CC, other PPs and packages.

10.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP;

b) the package(s) that the PP claims conformance to.

10.4.1.3 Action APE_CCL.1.1E

10.4.1.3.1 General

CC Part 3 APE_CCL.1.1C: *The conformance claim shall identify the CC edition to which the PP claims conformance.*

10.4.1.3.2 Work unit APE_CCL.1-1

The evaluator ***shall check*** that the conformance claim identifies the edition of the CC to which the PP claims conformance.

The evaluator determines that the CC conformance claim identifies the edition of the CC that was used to develop this PP. This should include the edition number of the CC and, unless the International English edition of the CC was used, the language of the edition of the CC that was used.

CC Part 3 APE_CCL.1.2C: *The conformance claim shall describe the conformance of the PP to CC Part 2 as either CC Part 2 conformant or CC Part 2extended.*

10.4.1.3.3 Work unit APE_CCL.1-2

The evaluator ***shall check*** that the CC conformance claim states a claim of either CC Part 2 conformant or CC Part 2 extended for the PP.

CC Part 3 APE_CCL.1.3C: *The conformance claim shall describe the conformance of the PP as either "CC Part 3 conformant" or "CC Part 3 extended".*

10.4.1.3.4 Work unit APE_CCL.1-3

The evaluator ***shall check*** that the CC conformance claim states a claim of either CC Part 3 conformant or CC Part 3 extended for the PP.

CC Part 3 APE_CCL.1.4C: *The conformance claim shall be consistent with the extended components definition.*

10.4.1.3.5 Work unit APE_CCL.1-4

The evaluator ***shall examine*** the CC conformance claim for CC Part 2 to determine that it is consistent with the extended components definition.

If the CC conformance claim contains CC Part 2 conformant, the evaluator determines that the extended components definition does not define functional components.

If the CC conformance claim contains CC Part 2 extended, the evaluator determines that the extended components definition defines at least one extended functional component.

10.4.1.3.6 Work unit APE_CCL.1-5

The evaluator ***shall examine*** the CC conformance claim for CC Part 3 to determine that it is consistent with the extended components definition.

If the CC conformance claim contains CC Part 3 conformant, the evaluator determines that the extended components definition does not define assurance components.

If the CC conformance claim contains CC Part 3 extended, the evaluator determines that the extended components definition defines at least one extended assurance component.

CC Part 3 APE_CCL.1.5C: *The conformance claim shall identify all PPs and packages to which the PP claims conformance.*

10.4.1.3.7 Work unit APE_CCL.1-6

The evaluator ***shall check*** that the conformance claim contains a PP claim that identifies all PPs for which the PP claims conformance.

If the PP does not claim conformance to another PP, this work unit is not applicable and therefore considered to be satisfied.

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 conformance to a PP are not permitted.

If the PP is of exact conformance type, the evaluator determines that the PP claims conformance to no other PPs. If the PP is not of exact conformance type, the evaluator determines that any PPs to which conformance is claimed do not require exact conformance in their conformance statement.

10.4.1.3.8 Work unit APE_CCL.1-7

The evaluator ***shall check***, for each identified functional package, that the package definition is complete.

If the PP does not claim conformance to a functional package, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 "Packages" by checking that the functional package includes:

a) A functional package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) A functional package overview, giving a narrative description of the security functionality;

c) A component rationale that provides the rationale for selecting the functional components/requirements included in the package;

d) If the package defines an SPD then:

i. The package includes a security requirements rationale.

ii. The package includes a security objectives rationale if security objectives for the environment are defined.

iii. If the package is a direct rationale package, there are no security objectives for the TOE defined and the security requirements rationale maps directly to the SPD.

iv. If the package is not a direct rationale package, then security objectives for the TOE are defined, the security objectives rationale covers the objectives with respect to the SPD, and the security requirements rationale maps the requirements to the security objectives.

e) one or more security components or requirements (the functional package SFRs);

f) If extended components have been specified then the functional package includes an extended components definition;

10.4.1.3.9 Work unit APE_CCL.1-8

The evaluator ***shall check***, for each identified assurance package, that the package definition is complete. If the PP does not claim conformance to an assurance package, this work unit is not applicable and therefore considered to be satisfied. If the assurance package is a reference to one of the assurance packages contained in CC Part 5 then this work unit is also considered to be satisfied. The evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 "Packages" by checking that the assurance package includes:

a) An assurance package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) An assurance package overview, giving a narrative description of the security functionality;

c) One or more security components or requirements (the assurance package SARs) drawn from CC Part 3, extended assurance components or some combination of both;

d) An assurance package shall not include an SPD or security objectives;

e) If extended components have been specified then the assurance package includes an extended components definition;

f) A security requirements rationale that provides the rationale for selecting the assurance components/ requirements included in the package.

CC Part 3 APE_CCL.1.6C: *The conformance claim shall describe any conformance of the PP to a functional package as one of package-conformant, package-augmented, or package-tailored.*

10.4.1.3.10 Work unit APE_CCL.1-9

The evaluator ***shall check*** that, for each identified functional package, the conformance claim states a claim of conformance to that package as one of package-conformant, package-augmented, or package-tailored. If the PP does not claim conformance to a functional package, this work unit is not applicable and therefore considered to be satisfied.

If the package conformance claim contains package-conformant, the evaluator determines that all assumptions, threats, OSPs, security objectives and SFRs included in the package are included

in identical form in the PP (after allowing for iteration, refinement, assignments and selections from the package to be completed as required by the PP).

If the package conformance claim contains package-augmented, the evaluator determines that all assumptions, threats, OPSs, Security Objectives, and SFRs included in the package are included in identical form in the PP (after allowing for iteration, refinement, assignments and selections from the package to be completed as required by the PP) except that the PP shall have at least one additional SFR or one SFR that is hierarchically higher than an SFR in the functional package.

If the package conformance claim contains package-tailored, the evaluator determines that all assumptions, threats, OPSs, Security Objectives, and SFRs included in the package are included in identical form in the PP (after allowing for iteration, refinement, assignments and selections from the package to be completed as required by the PP) except that the PP shall include additional selection values for at least one of the SFRs in the package, and may also have one or more SFRs that are hierarchically higher than an SFR in the functional package.

CC Part 3 APE_CCL.1.7C: *The conformance claim shall describe any conformance of the PP to an assurance package as either package-conformant or package-augmented.*

10.4.1.3.11 Work unit APE_CCL.1-10

The evaluator ***shall check*** that, for each identified assurance package, the conformance claim states a claim of conformance to that package as one of package-conformant or package-augmented.

If the package conformance claim contains package-conformant, the evaluator determines that the PP contains all SARs included in the package, but no additional SARs.

If the package conformance claim contains package-augmented, the evaluator determines that the PP contains all SARs included in the package, and at least one additional SAR or at least one SAR that is hierarchical to a SAR in the package.

CC Part 3 APE_CCL.1.8C: *The conformance claim shall describe any conformance of the PP to another PP as PP Conformant.*

10.4.1.3.12 Work unit APE_CCL.1-11

The evaluator ***shall check*** that if the PP claims conformance to another PP, that claim is described as PP Conformant.

If the PP does not claim conformance to another PP, this work unit is considered satisfied.

CC Part 3 APE_CCL.1.9C: *The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PP(s) to which conformance is being claimed.*

10.4.1.3.13 Work unit APE_CCL.1-12

The evaluator ***shall examine*** the conformance claim rationale to determine that the TOE type of the TOE is consistent with all TOE types of the PPs.

If the PP does not claim conformance to another PP, this work unit is not applicable and therefore considered to be satisfied.

The relation between the types may be simple: a firewall PP claiming conformance to another firewall PP, or more complex: a smart card PP claiming conformance to a number of other PPs at the same time: a PP for the integrated circuit, a PP for the smart card OS, and two PPs for two applications on the smart card.

CC Part 3 APE_CCL.1.10C: *The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs and any functional packages for which conformance is being claimed.*

10.4.1.3.14 Work unit APE_CCL.1-13

The evaluator ***shall examine*** the conformance claim rationale to determine that it demonstrates that the statement of security problem definition is consistent with the statements of security problem definition stated in the PPs and functional packages to which conformance is being claimed, taking into consideration the conformance statement of PPs to which conformance is being claimed.

If the PP under evaluation does not claim conformance with any PP or functional packages, this work unit is not applicable and therefore considered to be satisfied.

If the conformance claim includes functional packages, the evaluator determines that the security problem definition of the PP under evaluation consists of all assumptions, threats and OSPs of all functional packages.

The terms exact, strict and demonstrable conformance are defined in CC Part 1.

If packages are used, the rules defined in the following paragraphs concerning strict and demonstrable conformance also hold for the SPD descriptions taken from the packages.

Note that since a PP that specifies strict or demonstrable conformance in its conformance statement can only claim conformance to another PP whose conformance statement requires strict or demonstrable conformance, those are the only cases covered in the following paragraphs. If strict conformance is required by the PP to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether:

a) the threats in the PP under evaluation are a superset of or identical to the threats in the PP to which conformance is being claimed;

b) the OSPs in the PP under evaluation are a superset of or identical to the OSPs in the PP to which conformance is being claimed;

c) the assumptions in the PP claiming conformance are identical to the assumptions in the PP to which conformance is being claimed, with two possible exceptions described in the following two bullet points:

   - an assumption (or part of an assumption) from the PP to which conformance is claimed, can be omitted, if all security objectives for the operational environment addressing this assumption (or part of an assumption) are replaced by security objectives for the TOE;

   - an assumption can be added to the assumptions defined in the PP to which conformance is claimed, if a justification is given, why the new assumption neither mitigates a threat (or a part of a threat) meant to be addressed by security objectives for the TOE in the PP to which conformance is claimed, nor fulfils an OSP (or part of an OSP) meant to be addressed by security objectives for the TOE in the PP to which conformance is claimed.

The following discussion gives some motivation and examples for cases where a PP, omits assumptions from another PP to which conformance is claimed, or adds new assumptions:

- example for omitting an assumption: A PP to which conformance is claimed, may contain an assumption stating that the operational environment prevents unauthorized modification or interception of data sent to an external interface of the TOE. This may be the case if the TOE accepts data in clear text and without integrity protection at this interface and is assumed to be located in a secure operational environment, which will prevent attackers from accessing these data. The assumption will then be mapped in the PP, to which conformance is claimed, to some objective for the operational environment stating that the data interchanged at this interface are protected by adequate measures in the operational environment. If a PP claiming this PP, defines a more secure TOE, which has an additional security objective stating that the TOE itself protects these data, for example by providing a secure channel for encryption and integrity protection of all data transferred via this interface, the corresponding objective and assumption for the operational environment can be omitted from the PP claiming conformance. This is also called re-assigning of the objective, since the objective is re-assigned from the operational environment to the TOE. Note, that this TOE is still secure in an operational environment fulfilling the omitted assumption and therefore still fulfils the PP to which conformance is claimed.

- example for adding an assumption: In this example, the PP to which conformance is claimed, is designed to specify requirements for a TOE of type "Firewall" and the author of another PP wishes to claim conformance to this PP for a TOE, which implements a firewall, but additionally provides the functionality of a virtual private network (VPN) component. For the VPN functionality, the TOE needs cryptographic keys and these keys may also have to be handled securely by the operational environment (e. g. if symmetric keys are used to secure the network connection and therefore need to be provided in some secure way to other components in the network). In this case, it is acceptable to add an assumption that the cryptographic keys used by the VPN are handled securely by the operational environment. This assumption does not address threats or OSPs of the PP to which conformance is claimed, and therefore fulfils the conditions stated above.

- counterexample for adding an assumption: In a variant of the first example a PP to which conformance is claimed, may already contain an objective for the TOE to provide a secure channel for one of its interfaces, and this objective is mapped to a threat of unauthorized modification or reading of the data on this interface. In this case, it is clearly not allowed for another PP claiming this PP, to add an assumption for the operational environment, which assumes that the operational environment protects data on this interface against modification or unauthorized reading of the data. This assumption would reduce a threat, which is meant to be addressed by the TOE. Therefore, a TOE fulfilling a PP with this added assumption would not automatically fulfil the PP to which conformance is claimed, anymore and this addition is therefore not allowed.

- second counterexample for adding an assumption: In the example above of a TOE implementing a firewall it would not be admissible to add a general assumption that the TOE is only connected to trusted devices, because this would obviously remove essential threats relevant for a firewall (namely that there is untrusted IP traffic, which needs to be filtered). Therefore, this addition would not be allowed.

If demonstrable conformance is required by the PP to which conformance is being claimed, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security problem definition of the PP under evaluation is equivalent or more restrictive than the statement of security problem definition in the PP to which conformance is being claimed.

For this, the conformance claim rationale needs to demonstrate that the security problem definition in the PP claiming conformance is equivalent (or more restrictive) than the security problem definition in the PP to which conformance is claimed. This means that:

- all TOEs that would meet the security problem definition in the PP claiming conformance also meet the security problem definition in the PP to which conformance is claimed. This can also be shown indirectly by demonstrating that every event, which realizes a threat defined in the PP to which conformance is claimed, or violates an OSP defined in the PP to which conformance is claimed, would also realize a threat stated in the PP claiming conformance or violate an OSP defined in the PP claiming conformance. Note that fulfilling an OSP stated in the PP claiming conformance may avert a threat stated in the PP to which conformance is claimed, or that averting a threat stated in the PP claiming conformance may fulfil an OSP stated in the PP to which conformance is claimed, so threats and OSPs can substitute each other;

- all operational environments that would meet the security problem definition in the PP to which conformance is claimed, would also meet the security problem definition in the PP claiming conformance (with one exception in the next bullet);

- besides a set of assumptions in the PP claiming conformance needed to demonstrate conformance to the SPD of the PP to which conformance is claimed, a PP claiming conformance may specify further assumptions, but only if these additional assumptions are independent of and do not affect the security problem definition as defined in the PP to which conformance is claimed. More detailed, there are no assumptions in the PP claiming conformance that exclude threats to the TOE that need to be countered by the TOE according to the PP to which conformance is claimed. Similarly, there are no assumptions in the PP claiming conformance that realize aspects of an OSP stated in the PP to which conformance is claimed, which are meant to be fulfilled by the TOE according to the PP to which conformance is claimed.

CC Part 3 APE_CCL.1.11C: *The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PPs and any functional packages for which conformance is being claimed.*

10.4.1.3.15 Work unit APE_CCL.1-14

The evaluator ***shall examine*** the conformance claim rationale to determine that the statement of security objectives is consistent with the statement of security objectives in the PPs and functional packages to which conformance is being claimed, taking into consideration the conformance statement of PPs to which conformance is being claimed..

If the PP does not claim conformance to another PP nor to functional packages, this work unit is not applicable and therefore considered to be satisfied.

If conformance is claimed to any functional packages, the evaluator determines that the security objectives of the PP under evaluation include all security objectives of all functional packages to which conformance is claimed.

If packages are used, the rules defined in the following paragraphs concerning strict and demonstrable conformance also hold for the security objectives taken from the packages.

Note that since a PP can only claim conformance to another PP whose conformance statement requires strict or demonstrable conformance, those are the only cases covered in the following paragraphs. If strict conformance is required by the PP to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether:

a) the PP under evaluation contains all security objectives for the TOE of the PP to which conformance is being claimed. Note that it is allowed for the PP under evaluation to have additional security objectives for the TOE;

b) the security objectives for the operational environment in the PP claiming conformance are identical to the security objectives for the operational environment in the PP to which conformance is being claimed, with two possible exceptions described in the following two bullet points;

c) a security objective for the operational environment (or part of such security objective) from the PP to which conformance is claimed, can be replaced by the same (part of the) security objective stated for the TOE;

d) a security objective for the operational environment can be added to the objectives defined in the PP to which conformance is claimed, if a justification is given, why the new objective neither mitigates a threat (or a part of a threat) meant to be addressed by security objectives for the TOE in the PP to which conformance is claimed, nor fulfils an OSP (or part of an OSP) meant to be addressed by security objectives for the TOE in the PP to which conformance is claimed.

When examining a PP claiming another PP which omits security objectives for the operational environment from the PP to which conformance is claimed, or adds new security objectives for the operational environment, the evaluator shall carefully determine, if the conditions given above are fulfilled. The examples given for the case of assumptions in the preceding work unit are also valid here.

If demonstrable conformance is required by the PP to which conformance is being claimed, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security objectives of the PP under evaluation is equivalent or more restrictive than the statement of security objectives in the PP to which conformance is being claimed.

For this the conformance claim rationale needs to demonstrate that the security objectives in the PP claiming conformance are equivalent to (or more restrictive than) the security objectives in the PP to which conformance is claimed. This means that:

a) all TOEs that would meet the security objectives for the TOE in the PP claiming conformance also meet the security objectives for the TOE in the PP to which conformance is claimed;

b) all operational environments that would meet the security objectives for the operational environment in the PP to which conformance is claimed, would also meet the security objectives for the operational environment in the PP claiming conformance (with one exception in the next bullet);

c) besides a set of security objectives for the operational environment in the PP claiming conformance, which are used to demonstrate conformance to the set of security objectives defined in the PP to which conformance is claimed, a PP claiming conformance may specify further security objectives for the operational environment, but only if these security objectives neither affect the original set of security objectives for the TOE nor the security objectives for the operational environment as defined in the PP to which conformance is claimed.

CC Part 3 APE_CCL.1.12C: *The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PPs, and any functional packages for which conformance is being claimed.*

10.4.1.3.16 Work unit APE_CCL.1-15

The evaluator ***shall examine*** the PP to determine that the statement of security requirements is consistent with all security requirements stated in the PPs and any functional packages to which conformance is being claimed, taking into consideration the conformance statements of PPs to which conformance is being claimed.

If the PP does not claim conformance to another PP nor to functional packages, this work unit is not applicable and therefore considered to be satisfied.

If conformance is claimed to any functional packages, the evaluator determines that the SFRs of the PP under evaluation include all SFRs (or hierarchical SFRs) of all functional packages to which conformance is claimed.

If packages are used, the rules defined in the following paragraphs concerning strict and demonstrable conformance also hold for the SFRs taken from the packages.

Note that since a PP can only claim conformance to another PP whose conformance statement requires strict or demonstrable conformance, those are the only cases covered in the following paragraphs.

If strict conformance is required by the PP to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether the statement of security requirements in the PP under evaluation is a superset of or identical to the statement of security requirements in the PP to which conformance is being claimed (for strict conformance).

If demonstrable conformance is required by the PP to which conformance is being claimed, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security requirements of the PP under evaluation is equivalent or more restrictive than the statement of security requirements in the PP to which conformance is being claimed.

For:

- SFRs: The conformance rationale in the PP claiming conformance shall demonstrate that the overall set of requirements defined by the SFRs in the PP claiming conformance is equivalent (or more restrictive) than the overall set of requirements defined by the SFRs in the PP to which conformance is claimed. This means that all TOEs that would meet the requirements defined by the set of all SFRs in the PP claiming conformance would also meet the requirements defined by the set of all SFRs in the PP to which conformance is claimed;

- SARs: The PP claiming conformance shall contain all SARs in the PP to which conformance is claimed, but may claim additional SARs or replace SARs by hierarchically stronger SARs. The completion of operations in the PP claiming conformance must be consistent with that in the PP to which conformance is claimed; either the same completion will be used in the PP claiming conformance as that in the PP to which conformance is claimed or a completion that makes the SAR more restrictive (the rules of refinement apply).

CC Part 3 APE_CCL.1.13C: *The conformance statement shall describe the conformance required of any PPs/STs to the PP as exact-PP, strict-PP or demonstrable-PP conformance.*

10.4.1.3.17 Work unit APE_CCL.1-16

The evaluator ***shall check*** that the PP conformance statement states a claim of exact-PP, strict-PP or demonstrable-PP conformance.

CC Part 3 APE_CCL.1.14C: *For an exact conformance PP, the conformance statement shall contain an allowed-with statement that identifies the set of PPs (if any) to which, in combination with the PP under evaluation, exact conformance is allowed to be claimed.*

10.4.1.3.18 Work unit APE_CCL.1-17

The evaluator ***shall check*** the conformance statement to determine that it contains an allowed-with statement that lists the set of PPs to which, in combination with the PP being evaluated, an exact conformance claim (in an ST or PP-Configuration) is allowed.

If the PP does not require exact conformance in its conformance statement, this work unit does not apply and is therefore considered satisfied.

If the PP does not allow claims of exact conformance to it in combination with any other PPs, then no list of PPs is required and this work unit is considered satisfied.

The usage of the allowed-with statement is a matter of policy and is not addressed here. There are no other actions for the evaluator other than determining that the list is present.

CC Part 3 APE_CCL.1.15C: *For an exact conformance PP, the conformance statement shall contain an allowed-with statement that identifies the set of PP-Modules (if any) that are allowed to be used with the PP under evaluation in a PP-Configuration.*

10.4.1.3.19 Work unit APE_CCL.1-18

The evaluator ***shall check*** the conformance statement to determine that it contains an allowed-with statement that lists the set of PP-Modules that, in combination with the PP being evaluated, can be present in a PP-Configuration that requires exact conformance.

If the PP does not require exact conformance in its conformance statement, this work unit does not apply and is therefore considered satisfied.

If the PP does not allow claims of exact conformance to it in combination with any PP-Modules, then no list of PP-Modules is required and this work unit is considered satisfied.

The usage of the allowed-with statement is a matter of policy and is not addressed here. There are no other actions for the evaluator other than determining that the list is present.

CC Part 3 APE_CCL.1.16C: *The conformance statement shall identify the set of derived Evaluation Methods and Evaluation Activities (if any) that shall be used with the PP under evaluation. This list shall contain:*

- *any Evaluation Methods and Evaluation Activities that are specified for the PP under evaluation;*
- *any Evaluation Methods and Evaluation Activities specified in conformance statements of PPs to which conformance is being claimed by the PP under evaluation;*
- *any Evaluation Methods and Evaluation Activities specified in the Security Requirements sections of packages to which conformance is being claimed by the PP under evaluation.*

10.4.1.3.20 Work unit APE_CCL.1-19

The evaluator ***shall check*** the conformance statement of the PP under evaluation to determine that:

a) if any derived Evaluation Methods and Evaluation Activities are required by other items to which the PP claims conformance then these are all identified in the PP under evaluation, along with any derived Evaluation Methods and Evaluation Activities that the PP itself requires;

b) the list of derived Evaluation Methods and Evaluation Activities is sufficiently structured and detailed to unambiguously identify and locate every member of the list;

c) if there is any overlap in the scope of the identified Evaluation Methods and Evaluation Activities (i.e. where an overlay exists as described in CC Part 4) then the rationale for the resulting set of Evaluation Methods and Evaluation Activities is applicable to the TOE as described by the PP under evaluation.

The intention of this work unit is to ensure that when evaluating a TOE that claims conformance with the PP under evaluation then the correct Evaluation Methods and Evaluation Activities can be used. This means that identification in the PP need not list individual Evaluation Activities where these are unambiguously included in a listed Evaluation Method. Similarly, where multiple Evaluation Methods or Evaluation Activities are included in a single document then it is sufficient to reference the document, provided this does lead to unambiguous identification of the Evaluation Methods and Evaluation Activities that apply to the PP under evaluation.

EXAMPLE If a document lists multiple different Evaluation Methods applicable to different use cases then it would not be sufficient to reference the document: the relevant use cases would also have to be identified.

10.5 Security problem definition (APE_SPD)

10.5.1 Evaluation of sub-activity (APE_SPD.1)

10.5.1.1 Objectives

The objective of this sub-activity is to determine that the security problem intended to be addressed by the TOE and its operational environment is clearly defined.

10.5.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP.

10.5.1.3 Action APE_SPD.1.1E

10.5.1.3.1 General

CC Part 3 APE_SPD.1.1C: *The security problem definition shall describe the threats.*

10.5.1.3.2 Work unit APE_SPD.1-1

The evaluator ***shall check*** that the security problem definition describes the threats.

If all security objectives are derived from assumptions and/or OSPs 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.

The evaluator determines that the security problem definition describes the threats that must be countered by the TOE and/or its operational environment.

Note that if optional requirements are defined by the PP, there may be associated threats that are covered by this work unit.

CC Part 3 APE_SPD.1.2C: *All threats shall be described in terms of a threat agent, an asset, and an adverse action.*

10.5.1.3.3 Work unit APE_SPD.1-2

The evaluator ***shall examine*** the security problem definition to determine that all threats are described in terms of a threat agent, an asset, and an adverse action.

If all security objectives are derived from assumptions and OSPs 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.

Threat agents may be further described by aspects such as expertise, resource, opportunity, and motivation.

CC Part 3 APE_SPD.1.3C: *The security problem definition shall describe the organizational security policies (OSPs).*

10.5.1.3.4 Work unit APE_SPD.1-3

The evaluator ***shall examine*** that the security problem definition describes the OSPs.

If all security objectives are derived from assumptions and/or threats only, OSPs need not be present in the PP. In this case, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that OSP statements are made in terms of rules or guidelines that must be followed by the TOE and/or its operational environment.

The evaluator determines that each OSP 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.

Note that if optional requirements are defined by the PP, there may be associated OSPs that are covered by this work unit.

CC Part 3 APE_SPD.1.4C: *The security problem definition shall describe the assumptions about the operational environment of the TOE.*

10.5.1.3.5 Work unit APE_SPD.1-4

The evaluator ***shall examine*** the security problem definition to determine that it describes the assumptions about the operational environment of the TOE.

If there are no assumptions, this work unit is not applicable and is therefore considered to be satisfied.

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

10.6 Security objectives (APE_OBJ)

10.6.1 Evaluation of sub-activity (APE_OBJ.1)

10.6.1.1 Objectives

The objective of this sub-activity is to determine whether the security objectives for the operational environment are clearly defined.

10.6.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP.

10.6.1.3 Action APE_OBJ.1.1E

10.6.1.3.1 General

CC Part 3 APE_OBJ.1.1C: *The statement of security objectives shall describe the security objectives for the operational environment.*

10.6.1.3.2 Work unit APE_OBJ.1-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the operational environment.

The evaluator checks that the security objectives for the operational environment are identified.

CC Part 3 APE_OBJ.1.2C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

10.6.1.3.3 Work unit APE_OBJ.1-2

The evaluator ***shall check*** that the security objectives rationale traces the security objectives for the operational environment back to threats countered by that security objective, to OSPs enforced by that security objective, and to assumptions upheld by that security objective.

Each security objective for the operational environment may trace back to threats, OSPs, assumptions, or a combination of threats, OSPs and/or assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace a security objective for the operational environment back to at least a threat, OSP or assumption, implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

There is no prescribed location where this tracing element of the rationale must be placed: for example, the relevant parts may be located under each threat, OSP and assumption in order to help make the security argument clearer and easier to read.

CC Part 3 APE_OBJ.1.3C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

10.6.1.3.4 Work unit APE_OBJ.1-3

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that simply listing in the security objectives rationale the security objectives for the operational environment associated with each assumption may be a part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "Security Objective X directly upholds Assumption Y".

10.6.2 Evaluation of sub-activity (APE_OBJ.2)

10.6.2.1 Objectives

The objective of this sub-activity is to determine whether the security objectives adequately and completely address the security problem definition and that the division of this problem between the TOE and its operational environment is clearly defined.

10.6.2.2 Input

The evaluation evidence for this sub-activity is:

a) the PP.

10.6.2.3 Action APE_OBJ.2.1E

10.6.2.3.1 General

CC Part 3 APE_OBJ.2.1C: *The statement of security objectives shall describe the security objectives for the TOE and the security objectives for the operational environment.*

10.6.2.3.2 Work unit APE_OBJ.2-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the TOE and the security objectives for the operational environment.

The evaluator checks that both categories of security objectives are clearly identified and separated from the other category.

CC Part 3 APE_OBJ.2.2C: *The security objectives rationale shall trace each security objective for the TOE back to threats countered by that security objective and OSPs enforced by that security objective.*

10.6.2.3.3 Work unit APE_OBJ.2-2

The evaluator ***shall check*** that the security objectives rationale traces all security objectives for the TOE back to threats countered by the objectives and/or OSPs enforced by the objectives.

Each security objective for the TOE may trace back to threats or OSPs, or a combination of threats and OSPs, but it must trace back to at least one threat or OSP. Optional requirements may require Threats/OSPs to be specified, and security objectives associated with these SPD elements are also covered by this work unit.

Failure to trace a security objective for the TOE back to at least one threat or OSP, implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the TOE has no useful purpose.

CC Part 3 APE_OBJ.2.3C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

10.6.2.3.4 Work unit APE_OBJ.2-3

The evaluator ***shall check*** that the security objectives rationale traces the security objectives for the operational environment back to threats countered by that security objective, to OSPs enforced by that security objective, and to assumptions upheld by that security objective.

Each security objective for the operational environment may trace back to threats, OSPs, assumptions, or a combination of threats, OSPs and/or assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace a security objective for the operational environment back to at least one threat, OSP, or assumption, implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

CC Part 3 APE_OBJ.2.4C: *The security objectives rationale shall demonstrate that the security objectives counter all threats.*

10.6.2.3.5 Work unit APE_OBJ.2-4

The evaluator ***shall examine*** the security objectives rationale to determine that it justifies for each threat that the security objectives are suitable to counter that threat.

If no security objectives trace back to the threat, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

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

Note that the tracings from security objectives to threats provided in the security objectives rationale may be 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 may be as minimal as "Security Objective X directly counters Threat Y".

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

CC Part 3 APE_OBJ.2.5C: *The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.*

10.6.2.3.6 Work unit APE_OBJ.2-5

The evaluator ***shall examine*** the security objectives rationale to determine that for each OSP it justifies that the security objectives are suitable to enforce that OSP.

If no security objectives trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all security objectives that trace back to that OSP are achieved, the OSP is enforced.

The evaluator also determines that each security objective that traces back to an OSP is necessary: when the security objective is achieved it actually contributes to the enforcement of the OSP.

Note that the tracings from security objectives to OSPs provided in the security objectives rationale may be part of a justification, but do not constitute a justification by themselves. In the case that a security objective is merely a statement reflecting the intent to enforce a particular OSP, a justification is required, but this justification may be as minimal as "Security Objective X directly enforces OSP Y".

CC Part 3 APE_OBJ.2.6C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

10.6.2.3.7 Work unit APE_OBJ.2-6

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that the tracings from security objectives for the operational 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 operational environment is merely a restatement of an assumption, a justification is required, but this justification may be as minimal as "Security Objective X directly upholds Assumption Y".

10.7 Extended components definition (APE_ECD)

10.7.1 Evaluation of sub-activity (APE_ECD.1)

10.7.1.1 Objectives

The objective of this sub-activity is to determine whether extended components have been clearly and unambiguously defined, and whether they are necessary, i.e. they may not be clearly expressed using existing CC Part 2 or CC Part 3 components.

10.7.1.2 Input

The evaluation evidence for this sub-activity is the PP.

10.7.1.3 Action APE_ECD.1.1E

10.7.1.3.1 General

CC Part 3 APE_ECD.1.1C: *The statement of security requirements shall identify all extended security requirements.*

10.7.1.3.2 Work unit APE_ECD.1-1

The evaluator ***shall check*** that all security requirements in the statement of security requirements that are not identified as extended requirements are present in CC Part 2 or in CC Part 3.

CC Part 3 APE_ECD.1.2C: *The extended components definition shall define an extended component for each extended security requirement.*

10.7.1.3.3 Work unit APE_ECD.1-2

The evaluator ***shall check*** that the extended components definition defines an extended component for each extended security requirement.

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

A single extended component may be used to define multiple iterations of an extended security requirement, it is not necessary to repeat this definition for each iteration.

CC Part 3 APE_ECD.1.3C: *The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.*

10.7.1.3.4 Work unit APE_ECD.1-3

The evaluator ***shall examine*** the extended components definition to determine that it describes how each extended component fits into the existing CC components, families, and classes.

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

The evaluator determines that each extended component is either:

a) a member of an existing CC Part 2 or CC Part 3 family; or

b) a member of a new family defined in the PP.

If the extended component is a member of an existing CC Part 2 or CC Part 3 family, the evaluator determines that the extended components definition adequately describes why the extended component should be a member of that family and how it relates to other components of that family.

If the extended component is a member of a new family defined in the PP, the evaluator confirms that the extended component is not appropriate for an existing family.

If the PP defines new families, the evaluator determines that each new family is either:

a) a member of an existing CC Part 2 or CC Part 3 class; or

b) a member of a new class defined in the PP.

If the family is a member of an existing CC Part 2 or CC Part 3 class, the evaluator determines that the extended components definition adequately describes why the family should be a member of that class and how it relates to other families in that class.

If the family is a member of a new class defined in the PP, the evaluator confirms that the family is not appropriate for an existing class.

10.7.1.3.5 Work unit APE_ECD.1-4

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended component identifies all applicable dependencies of that component.

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

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

CC Part 3 APE_ECD.1.4C: *The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.*

10.7.1.3.6 Work unit APE_ECD.1-5

The evaluator ***shall examine*** the extended components definition to determine that each extended functional component uses the existing CC Part 2 components as a model for presentation.

If the PP does not contain extended SFRs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended functional component is consistent with CC Part 2, 6.1.3, Component structure.

If the extended functional component uses operations, the evaluator determines that the extended functional component is consistent with CC Part 1, 8.2, Operations.

If the extended functional component is hierarchical to an existing functional component, the evaluator determines that the extended functional component is consistent with CC Part 2, 6.2.1, Component changes highlighting.

10.7.1.3.7 Work unit APE_ECD.1-6

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional family uses the existing CC functional families as a model for presentation.

If the PP does not define new functional families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional families are defined consistent with CC Part 2, 6.1.2, Family structure.

10.7.1.3.8 Work unit APE_ECD.1-7

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional class uses the existing CC functional classes as a model for presentation.

If the PP does not define new functional classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional classes are defined consistent with CC Part 2, 6.1.1, Class structure.

10.7.1.3.9 Work unit APE_ECD.1-8

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended assurance component uses the existing CC Part 3 components as a model for presentation.

If the PP does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended assurance component definition is consistent with CC Part 3, 6.1.3, Assurance component structure.

If the extended assurance component uses operations, the evaluator determines that the extended assurance component is consistent with CC Part 1, 8.2, Operations.

If the extended assurance component is hierarchical to an existing assurance component, the evaluator determines that the extended assurance component is consistent with CC Part 3, 6.1.3, Assurance component structure.

10.7.1.3.10 Work unit APE_ECD.1-9

The evaluator ***shall examine*** the extended components definition to determine that, for each defined extended assurance component, applicable methodology has been provided.

If the PP does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that, for each evaluator action element of each extended SAR, one or more work units are provided and that successfully performing all work units for a given evaluator action element will demonstrate that the element has been achieved.

10.7.1.3.11 Work unit APE_ECD.1-10

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance family uses the existing CC assurance families as a model for presentation.

If the PP does not define new assurance families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance families are defined consistent with CC Part 3, 6.1.2, Assurance family structure.

10.7.1.3.12 Work unit APE_ECD.1-11

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance class uses the existing CC assurance classes as a model for presentation.

If the PP does not define new assurance classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance classes are defined consistent with CC Part 3, 6.1.1, Assurance class structure.

CC Part 3 APE_ECD.1.5C: *The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements may be demonstrated.*

10.7.1.3.13 Work unit APE_ECD.1-12

The evaluator ***shall examine*** the extended components definition to determine that each element in each extended component is measurable and states objective evaluation requirements, such that conformance or nonconformance can be demonstrated.

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

The evaluator determines that elements of extended functional components are stated in such a way that they are testable, and traceable through the appropriate TSF representations.

The evaluator also determines that elements of extended assurance components avoid the need for subjective evaluator judgement.

The evaluator is reminded that whilst being measurable and objective is appropriate for all evaluation criteria, it is acknowledged that no formal method exists to prove such properties. Therefore, the existing CC functional and assurance components are to be used as a model for determining what constitutes conformance to this requirement.

10.7.1.4 Action APE_ECD.1.2E

10.7.1.4.1 Work unit APE_ECD.1-13

The evaluator ***shall examine*** the extended components definition to determine that each extended component may not be clearly expressed using existing components.

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

The evaluator should take components from CC Part 2 and CC Part 3, other extended components that have been defined in the PP, combinations of these components, and possible operations on these components into account when making this determination.

The evaluator is reminded that the role of this work unit is to preclude unnecessary duplication of components, that is, components that may be clearly expressed by using other components. The evaluator should not undertake an exhaustive search of all possible combinations of components including operations in an attempt to find a way to express the extended component by using existing components.

10.8 Security requirements (APE_REQ)

10.8.1 Evaluation of sub-activity (APE_REQ.1)

10.8.1.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs counter the threats and implement the organisational security policies of the TOE.

10.8.1.2 Input

The evaluation evidence for this sub-activity is the PP.

10.8.1.3 Action APE_REQ.1.1E

10.8.1.3.1 General

CC Part 3 APE_REQ.1.1C: *The statement of security requirements shall describe the SFRs and the SARs.*

10.8.1.3.2 Work unit APE_REQ.1-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2;

b) by reference to an extended component in the extended components definition of the PP;

c) by reference to a PP that the PP claims to be conformant with including any optional requirements defined in the PP;

d) by reference to a security requirements package that the PP claims to be conformant with;

e) by reproduction in the PP.

It is not required to use the same means of identification for all SFRs.

10.8.1.3.3 Work unit APE_REQ.1-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that each SAR is identified by one of the following means:

a) by reference to an individual component in CC Part 3;

b) by reference to an extended component in the extended components definition of the PP;

c) by reference to a PP that the PP claims to be conformant with;

d) by reference to a security requirements package that the PP claims to be conformant with;

e) by reproduction in the PP.

It is not required to use the same means of identification for all SARs.

CC Part 3 APE_REQ.1.2C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

10.8.1.3.4 Work unit APE_REQ.1-3

The evaluator ***shall examine*** the PP to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the PP defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the PP author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This can be especially applicable if the same terms are used in the rest of the PP.

CC Part 3 APE_REQ.1.3C: *The statement of security requirements shall identify all operations on the security requirements.*

10.8.1.3.5 Work unit APE_REQ.1-4

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

The evaluator determines that all operations are identified in each SFR or SAR where such an operation is used. This includes both completed operations and uncompleted operations. Identification may be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

If the PP defines *selection-based* SFRs, the evaluator determines that the PP clearly identifies the dependencies between the selection in an SFR and the selection-based SFR(s) to be included in the PP/ST should that selection be chosen by the PP/ST author.

CC Part 3 APE_REQ.1.4C: *All operations shall be performed correctly.*

10.8.1.3.6 Work unit APE_REQ.1-5

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.1.3.7 Work unit APE_REQ.1-6

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.1.3.8 Work unit APE_REQ.1-7

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.1.3.9 Work unit APE_REQ.1-8

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

CC Part 3 APE_REQ.1.5C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

10.8.1.3.10 Work unit APE_REQ.1-9

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that the security requirements rationale justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

- why the dependency is not necessary or useful, in which case no further information is required; or

- that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

The evaluator ensures that if a functional package defines a dependency on another functional package, then that functional package is included in the PP.

If a functional package identifies dependencies on its requirements that need to be satisfied by the underlying PP, the evaluator ensures that their analysis covers these dependencies as well.

CC Part 3 APE_REQ.1.6C: *The security requirements rationale shall trace each SFR back to the threats countered by that SFR and the OSPs enforced by that SFR.*

10.8.1.3.11 Work unit APE_REQ.1-10

The evaluator ***shall check*** that the security requirements rationale traces each SFR back to the threats countered by that SFR and OSPs enforced by that SFR.

The evaluator determines that each SFR is traced back to at least one threat or OSP for the TOE.

Failure to trace implies that either the security requirements rationale is incomplete, or the SFR has no useful purpose.

There is no prescribed location where this tracing element of the rationale must be placed: for example, the relevant parts may be located under each threat and OSP in order to help make the security argument clearer and easier to read.

CC Part 3 APE_REQ.1.7C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) counter all threats for the TOE.*

10.8.1.3.12 Work unit APE_REQ.1-11

The evaluator ***shall examine*** the security requirements rationale to determine that for each threat it demonstrates that the SFRs are suitable to meet that threat.

If no SFRs trace back to a threat, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

The evaluator determines that the justification for a threat demonstrates that the SFRs are sufficient: if all SFRs that trace back to the threat are achieved then, in the context of any applicable OSPs and assumptions, the threat is removed, sufficiently diminished, or the effects of the threat are sufficiently mitigated.

Note that simply listing in the security requirements rationale the SFRs associated with each threat may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly counters Threat Y".

The evaluator also determines that each SFR that traces back to a threat is necessary: when the SFR is implemented it actually contributes to the removal, diminishing or mitigation of that threat.

CC Part 3 APE_REQ.1.8C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) enforce all OSPs for the TOE.*

10.8.1.3.13 Work unit APE_REQ.1-12

The evaluator ***shall examine*** the security requirements rationale to determine that for each OSP it justifies that the SFRs are suitable to enforce that OSP.

If no SFRs or security objectives for the operational environment trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all SFRs that trace back to that OSP are achieved then, in the context of any applicable assumptions, the OSP is enforced.

The evaluator also determines that each SFR that traces back to an OSP is necessary: when the SFR is implemented it actually contributes to the enforcement of the OSP.

Note that simply listing in the security requirements rationale the SFRs associated with each OSP may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly enforces OSP Y".

CC Part 3 APE_REQ.1.9C: *The security requirements rationale shall explain why the SARs were chosen.*

10.8.1.3.14 Work unit APE_REQ.1-13

The evaluator ***shall check*** that the security requirements rationale explains why the SARs were chosen.

The evaluator is reminded that any explanation is correct, as long as it is coherent and neither the SARs nor the explanation have obvious inconsistencies with the remainder of the PP.

An example of an obvious inconsistency between the SARs and the remainder of the PP would be to have threat agents that are very capable, but an AVA_VAN SAR that does not protect against these threat agents.

CC Part 3 APE_REQ.1.10C: *The statement of security requirements shall be internally consistent.*

10.8.1.3.15 Work unit APE_REQ.1-14

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

- all optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

- all optional requirements are clearly identified as either:

    1) elective and therefore for inclusion purely at the discretion of the ST author (i.e. conformance to the PP does not depend on their inclusion in the ST); or

    2) conditional and therefore required if a conformant TOE implements the functionality covered by the requirement.

- all optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to "all objects", "all subjects" etc., that these requirements do not conflict.

Some possible conflicts are:

- an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended SAR specifying an open source review;

- FAU_Gen.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other

subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

- FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

- multiple iterations of FDP_ACC.1 Subset access control, especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

10.8.2 Evaluation of sub-activity (APE_REQ.2)

10.8.2.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs meet the security objectives of the TOE.

10.8.2.2 Input

The evaluation evidence for this sub-activity is the PP.

10.8.2.3 Action APE_REQ.2.1E

10.8.2.3.1 General

CC Part 3 APE_REQ.2.1C: *The statement of security requirements shall describe the SFRs and the SARs.*

10.8.2.3.2 Work unit APE_REQ.2-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2;

b) by reference to an extended component in the extended components definition of the PP;

c) by reference to an individual component in a PP that the PP claims to be conformant with, including any optional requirements defined in the PP;

d) by reference to an individual component in a security requirements package that the PP claims to be conformant with;

e) by reproduction in the PP.

It is not required to use the same means of identification for all SFRs.

10.8.2.3.3 Work unit APE_REQ.2-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that each SAR is identified by one of the following means:

a) by reference to an individual component in CC Part 3;

b) by reference to an extended component in the extended components definition of the PP;

c) by reference to an individual component in a PP that the PP claims to be conformant with;

d) by reference to an individual component in a security requirements package that the PP claims to be conformant with;

e) by reproduction in the PP.

It is not required to use the same means of identification for all SARs.

Note that if optional requirements are defined by the PP, there may be associated threats that are covered by this work unit.

CC Part 3 APE_REQ.2.2C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

10.8.2.3.4 Work unit APE_REQ.2-3

The evaluator ***shall examine*** the PP to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the PP defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the PP author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This can be especially applicable if the same terms are used in the rest of the PP.

CC Part 3 APE_REQ.2.3C: *The statement of security requirements shall identify all operations on the security requirements.*

10.8.2.3.5 Work unit APE_REQ.2-4

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

The evaluator determines that all operations are identified in each SFR or SAR where such an operation is used. This includes both completed operations and uncompleted operations. Identification may be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

If the PP defines *selection-based* SFRs, the evaluator determines that the PP clearly identifies the dependencies between the selection in an SFR and the selection-based SFR(s) to be included in the PP/ST should that selection be chosen by the PP/ST author.

CC Part 3 APE_REQ.2.4C: *All operations shall be performed correctly.*

10.8.2.3.6 Work unit APE_REQ.2-5

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.2.3.7 Work unit APE_REQ.2-6

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.2.3.8 Work unit APE_REQ.2-7

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

10.8.2.3.9 Work unit APE_REQ.2-8

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

CC Part 3 APE_REQ.2.5C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

10.8.2.3.10 Work unit APE_REQ.2-9

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that the security requirements rationale justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

a) why the dependency is not necessary or useful, in which case no further information is required; or

b) that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

The evaluator ensures that if a functional package defines a dependency on another functional package, then that functional package is included in the PP

If a functional package identifies dependencies on its requirements that need to be satisfied by the underlying PP, the evaluator ensures that their analysis covers these dependencies as well.

CC Part 3 APE_REQ.2.6C: *The security requirements rationale shall trace each SFR back to the security objectives for the TOE enforced by that SFR.*

10.8.2.3.11 Work unit APE_REQ.2-10

The evaluator ***shall check*** that the security requirements rationale traces each SFR back to the security objectives for the TOE.

Optional requirements may require Threats/OSPs to be specified, and security objectives associated with these SPD elements are also covered by this work unit.

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

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

CC Part 3 APE_REQ.2.7C: *The security requirements rationale shall demonstrate that the SFRs meet all security objectives for the TOE.*

10.8.2.3.12 Work unit APE_REQ.2-11

The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the TOE it justifies that the SFRs are suitable to meet that security objective for the TOE.

If no SFRs trace back to the security objective for the TOE, the evaluator action related to this work unit is assigned a fail verdict.

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

If the SFRs that trace back to a security objective for the TOE have any uncompleted assignments, or uncompleted or restricted selections, the evaluator determines that for every conceivable completion or combination of completions of these operations, the security objective is still met.

The evaluator also determines that each SFR that traces back to a security objective for the TOE is necessary: when the SFR is satisfied, it actually contributes to achieving the security objective.

Note that the tracings from SFRs 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.

CC Part 3 APE_REQ.2.8C: *The security requirements rationale shall explain why the SARs were chosen.*

10.8.2.3.13 Work unit APE_REQ.2-13

The evaluator ***shall check*** that the security requirements rationale explains why the SARs were chosen.

The evaluator is reminded that any explanation is correct, as long as it is coherent and neither the SARs nor the explanation have obvious inconsistencies with the remainder of the PP.

An example of an obvious inconsistency between the SARs and the remainder of the PP would be to have threat agents that are very capable, but an AVA_VAN SAR that does not protect against these threat agents.

CC Part 3 APE_REQ.2.9C: *The statement of security requirements shall be internally consistent.*

10.8.2.3.14 Work unit APE_REQ.2-14

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

- all optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

- all optional requirements are clearly identified as either:

    1) elective and therefore for inclusion purely at the discretion of the ST author (i.e. conformance to the PP does not depend on their inclusion in the ST); or

    2) conditional and therefore required if a conformant TOE implements the functionality covered by the requirement.

- all optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to "all objects", "all subjects" etc., that these requirements do not conflict.

Some possible conflicts are:

- an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended SAR specifying an open source review;

- FAU_GEN.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

- FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

- multiple iterations of FDP_ACC.1 Subset access control, especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

11 Class ACE: Protection Profile Configuration evaluation

11.1 General

The process of evaluating a PP-Configuration relies on the previous evaluation of all the PPs referenced in the PP-Configuration, and on the evaluation of each PP-Module, iteratively.

The ACE Class does not prescribe the order in which the PP-Modules are evaluated. A possible order is the following. Consider a PP-Configuration composed of a set of PPs: PP_1, … PP_n , and a set of PP-Modules: M_1, … M_k, where n ≥ 2 or k ≥ 1. Figure 7 presents the dependency structure of PPs and PP-Modules: PPs are standalone, i.e. they do not depend on other PPs or PP-Modules. If the PP-Configuration contains one or more PP-Modules, then a non-empty subset of these are based on PPs only (i.e. they do not depend on other PP-Modules). One possible evaluation order of such PP-Configuration consists of the following steps:

a) evaluation of the union of PPs (the PPs are assumed already evaluated following APE Class requirements);

b) if the PP-Configuration contains one or more PP-Modules, then follow the PP-Configuration's dependency structure from top (PPs) to bottom (PP-Modules) and evaluate all the components of each sub-set of PP-Modules, i.e. for each PP-Module:

    1) evaluate the PP-Module in the framework of its PP-Module Base (the elements of which have already been evaluated since the evaluation order meets the dependency structure) [Evaluation of sub-activity (ACE_MCO.1)];

    2) evaluate the PP-Module in the framework of all the PPs and already evaluated PP-Modules [contained in the same or previous subset of PP-Modules) (Evaluation of sub-activity (ACE_CCO.1)].

Due to the iterative nature of a PP-Configuration evaluation, it may be possible to mutualize the evaluation work to evaluate several configurations at the same time.

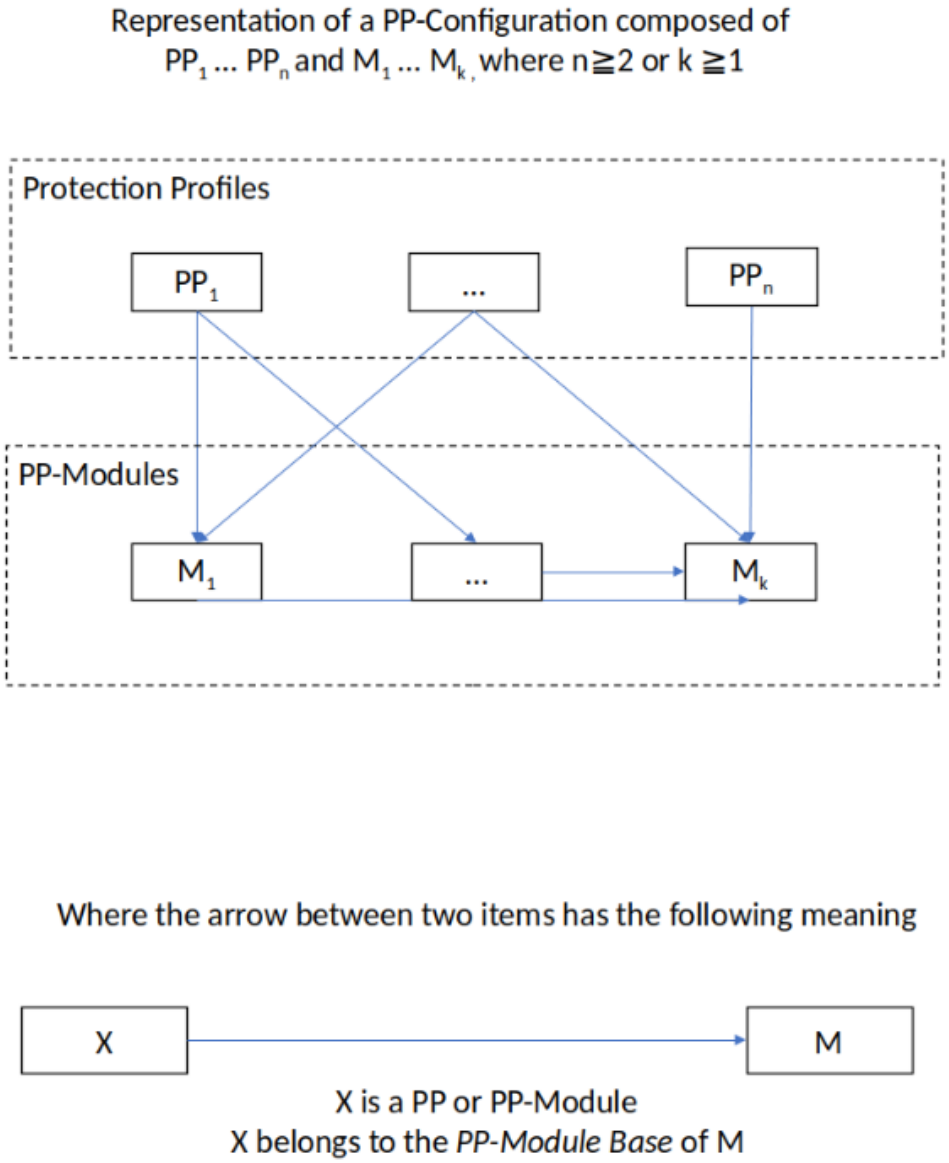

Figure 7 — Relationship between PPs and PP-Modules in a PP-Configuration

The ACE evaluation methodology is based on APE's.

11.2 PP-Module introduction (ACE_INT)

11.2.1 Evaluation of sub-activity (ACE_INT.1)

11.2.1.1 Objectives

The objective of this sub-activity is to determine whether the PP-Module is correctly identified, and whether the PP-Module Base and TOE overview are consistent with each other.

11.2.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Module;

b) its PP-Module Base.

11.2.1.3 Application notes

All actions of APE_INT.1.1E hold.

11.2.1.4 Action ACE_INT.1.1E

11.2.1.4.1 General

CC Part 3 ACE_INT.1.1C: *The PP-Module introduction shall contain a PP-Module reference, the identification of the PP-Module Base(s) and a TOE overview.*

11.2.1.4.2 Work unit ACE_INT.1-1

The evaluator ***shall check*** that the PP-Module introduction identifies the PP-Module Base(s) on which the PP-Module relies.

The evaluator ***shall check*** that the PP-Module introduction contains a PP-Module reference and a TOE overview.

CC Part 3 ACE_INT.1.2C: *The PP-Module reference shall uniquely identify the PP-Module.*

11.2.1.4.3 Work unit ACE_INT.1-2

The evaluator ***shall examine*** the PP-Module reference to determine that it uniquely identifies the PP-Module.

The evaluator determines that the PP-Module reference identifies the PP-Module itself, so that it may be easily distinguished from other PP-Modules, and that it also uniquely identifies each version of the PP-Module, e.g. by including a version number and/or a date of publication.

The PP-Module should have some referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates).

CC Part 3 ACE_INT.1.3C: *The identification of the PP-Module Base shall consist of at least one PP and possibly other PPs and PP-Modules on which the PP-Module depends.*

11.2.1.4.4 Work unit ACE_INT.1-3

The evaluator ***shall examine*** the PP-Module Base identification to determine that it includes a list of PPs and PP-Modules B_i of the following shape:

B_1 … *AND*… B_n with n≥ 1

The evaluator ***shall check*** that the set {B_1 ... B_n} contains one or more PPs.

For a PP-Module that allows to be used with alternative PP-Module Bases, the evaluator ***shall examine*** the PP-Module Base identification to determine that it includes multiple alternative lists of references that uniquely identify the sets S_i of PPs and PP-Modules on which the PP-Module may depend. The list of alternative PP-Module Bases has the following form:

S_1 ...*OR* ... S_k with k≥ 1

CC Part 3 ACE_INT.1.4C: *The identification of the PP-Module Base(s) shall describe the dependency structure of the PP-Module Base(s).*

11.2.1.4.5 Work unit ACE_INT.1-4

The evaluator ***shall examine*** the PP-Module Base identification to determine that it describes the dependency structure of the PPs and PP-Modules that compose the PP-Module Base.

Given the definition of a PP-Module Base Bi, the evaluator ***shall check*** that a PP-Module Base cannot consist solely of a PP-Module, and every PP-Module identifies its PP-Module Base.

The evaluator ***shall check*** that the PP-Module Base $\{B_1 ... B_n\}$ is a closed set, that is, for any PP-Module B_i, its own PP-Module Base belongs to the set $\{B_1 ... B_n\}$.

For a PP-Module that allows to be used with alternative PP-Module Bases, the evaluator ***shall check*** that each of those define a closed set of PPs and PP-Modules.

The evaluator ensures that if the PP-Module restricts the PP-Module Bases allowed for PP-Modules in the "being evaluated" PP-Module's PP-Module Base, those restrictions are stated in the introduction and are consistent for the depended-upon PP-Module's definition of allowed PP-Module Bases.

CC Part 3 ACE_INT.1.5C: *The PP-Module introduction shall contain as many TOE overviews as alternative PP-Module Bases.*

11.2.1.4.6 Work unit ACE_INT.1-5

The evaluator ***shall check*** that the PP-Module introduction contains as many TOE overviews as alternative PP-Module Bases. The statement of the TOE overview in a PP-Module may be given by reference when it is the same as in its PP-Module Base, i.e. when there is no addition.

If the TOE overview is not affected by the choice of alternative PP-Module Bases anywhere in the set of PP-Module Bases for the PP-Module, then a separate TOE description is not required and instead a note should be included to that effect.

CC Part 3 ACE_INT.1.6C: *The TOE overview shall summarize the usage and major security features of the TOE.*

11.2.1.4.7 Work unit ACE_INT.1-6

The evaluator ***shall examine*** the TOE overview to determine that it describes the usage and major security features of the TOE.

The TOE overview should briefly (i.e. several paragraphs) describe the usage and major security features expected of the TOE. The TOE overview should enable consumers and potential TOE developers to quickly determine whether the PP-Module is of interest to them.

The evaluator determines that the overview is clear enough for TOE developers and consumers, and sufficient to give them a general understanding of the intended usage and major security features of the TOE.

CC Part 3 ACE_INT.1.7C: *The TOE overview shall identify the TOE type.*

11.2.1.4.8 Work unit ACE_INT.1-7

The evaluator ***shall check*** that the TOE overview identifies the TOE type.

CC Part 3 ACE_INT.1.8C: *The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.*

11.2.1.4.9 Work unit ACE_INT.1-8

The evaluator ***shall examine*** the TOE overview to determine that it identifies any non-TOE hardware/software/firmware available to the TOE.

While some TOEs may run stand-alone, other TOEs (notably software TOEs) need additional hardware, software or firmware to operate. In this subclause of the PP-Module, the PP-Module author lists all hardware, software, and/or firmware that will be available for the TOE to run on.

This identification should be detailed enough for potential consumers and TOE developers to determine whether their TOE may operate with the listed hardware, software and firmware.

CC Part 3 ACE_INT.1.9C: *The TOE overview shall describe the differences of the TOE with regard to the TOEs defined in the PP-Module Base(s).*

11.2.1.4.10 Work unit ACE_INT.1-9

The evaluator ***shall check*** that the TOE overview identifies the differences introduced by the PP-Module with respect to the TOE overview of its PP-Module Base(s).

11.3 PP-Module conformance claims (ACE_CCL)

11.3.1 Evaluation of sub-activity (ACE_CCL.1)

11.3.1.1 Objectives

The objective of this sub-activity is to determine the validity of various conformance claims and of the conformance statement. These describe how the PP-Module conforms to CC Part 2 and packages. A PP-Module cannot claim conformance to any PP, PP-Configuration, or another PP-Module.

11.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Module;

b) the SFR and SAR package(s) that the PP-Module claims conformance to;

c) the PP-Configuration.

11.3.1.3 Action ACE_CCL.1.1E

CC Part 3 ACE_CCL.1.1C: *The conformance claim shall identify the CC edition to which the PP-Module claims conformance.*

11.3.1.3.1 Work unit ACE_CCL.1-1

The evaluator ***shall check*** that the conformance claim identifies the version of the CC to which the PP-Module claims conformance.

The evaluator determines that the CC conformance claim identifies the version of the CC that was used to develop this PP-Module. This should include the version number of the CC and, unless the English version of the CC was used, the language of the version of the CC that was used.

CC Part 3 ACE_CCL.1.2C: *The conformance claim shall describe the conformance of the PP-Module to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.*

11.3.1.3.2 Work unit ACE_CCL.1-2

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 2 conformant or CC Part 2 extended for the PP-Module.

CC Part 3 ACE_CCL.1.3C: *The conformance statement shall describe the conformance type required of any ST to the PP-Module (as part of a PP-Configuration) as one of exact, strict, or demonstrable.*

11.3.1.3.3 Work unit ACE_CCL.1-3

The evaluator ***shall check*** that the PP-Module conformance statement states a claim of exact, strict or demonstrable conformance which describes the manner in which STs shall conform to the PP-Module (as part of a PP-Configuration).

If the PP-Module states that strict conformance is required, STs shall conform to the PP-Module (as part of a PP-Configuration) in a strict manner.

If the PP-Module states that demonstrable conformance is required, STs shall conform to the PP-Module (as part of a PP-Configuration) in a strict or demonstrable manner.

If the PP-Module states that exact conformance is required, STs shall conform to the PP-Module (as part of a PP-Configuration) in an exact manner. In this case, the evaluator ***shall check*** that all of the PP-Module's referenced PP-Module Base(s) require exact conformance as well.

CC Part 3 ACE_CCL.1.4C: *The conformance claim shall describe the conformance of the PP-Module to this document as either "CC Part 3 conformant" or "CC Part 3 extended".*

11.3.1.3.4 Work unit ACE_CCL.1-4

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 3 conformant or CC Part 3 extended for the PP-Module.

CC Part 3 ACE_CCL.1.5C: *The conformance claim shall be consistent with the extended components definition.*

11.3.1.3.5 Work unit ACE_CCL.1-5

The evaluator ***shall examine*** the conformance claim for CC Part 2 and CC Part 3 to determine that it is consistent with the extended components definition.

If the CC conformance claim contains CC Part 2 and/or CC Part 3 conformant, the evaluator determines that the extended components definition does not define functional/assurance components.

If the CC conformance claim contains CC Part 2 and/or CC Part 3 extended, the evaluator determines that the extended components definition defines at least one extended functional/assurance component.

CC Part 3 ACE_CCL.1.6C: *The conformance claim shall identify all functional packages to which the PP-Module claims conformance.*

11.3.1.3.6 Work unit ACE_CCL.1-6

The evaluator ***shall check***, for each identified functional package, that the package definition is complete.

If the PP-Module does not claim conformance to a functional package, this work unit is not applicable and therefore considered to be satisfied.

For all functional packages that are not claimed by a component in the PP-Module's base, the evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 "Packages" by checking that the functional package includes:

a) A functional package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) A functional package overview, giving a narrative description of the security functionality;

c) A component rationale that provides the rationale for selecting the functional components/requirements included in the package;

d) If the package defines an SPD then:

   i. the package includes a security requirements rationale;

   ii. the package includes a security objectives rationale if security objectives for the environment are defined.;

   iii. if the package is a direct rationale package, there are no security objectives for the TOE defined and the security requirements rationale maps directly to the SPD;

   iv. If the package is not a direct rationale package, then security objectives for the TOE are defined, the security objectives rationale covers the objectives with respect to the SPD, and the security requirements rationale maps the requirements to the security objectives.

e) one or more security components or requirements (the functional package SFRs);

f) If extended components have been specified then the functional package includes an extended components definition;

CC Part 3 ACE_CCL.1.7C: *The conformance claim shall describe any conformance of the PP-Module to a functional package as either package-conformant, package-augmented or package-tailored.*

11.3.1.3.7 Work unit ACE_CCL.1-7

The evaluator ***shall check*** that, for each identified package, the conformance claim states a claim of either package-conformant, package-augmented or package-tailored.

The evaluator determines that for all functional packages claimed by the PP-Module that are already claimed by one of the PPs or PP-Modules in its PP-Module Base(s), the PP-Module augments or tailors the functional package as it is instantiated in its PP-Module Base (in this case the PP-Module would claim the functional package as "Package Augmented" or "Package Tailored" (as appropriate)). Otherwise, the evaluator confirms that functional packages claimed by a PP-Module Base component are not claimed by the PP-Module.

If the PP-Module does not claim conformance to a package, this work unit is not applicable and therefore considered to be satisfied.

If the functional package conformance claim contains package-conformant, the evaluator determines that all assumptions, threats, OSPs, security objectives and SFRs included in the package are included in identical form by the PP-Module (including via its PP-Module Base).

If the functional package conformance claim contains package-augmented, the evaluator determines that all assumptions, threats, OSPs, security objectives and SFRs included in the package are included in identical form by the PP-Module except that the PP-Module shall include additional selection items values for an at least one of the SFRs with existing selections in the

package, and may also have one or more SFRs that are hierarchically higher than an SFR in the functional package.

If the functional package conformance claim contains package-tailored, the evaluator determines that all assumptions, threats, OSPs, security objectives and SFRs included in the package are included in identical form by the PP-Module except that the PP-Module shall include additional selection values for at least one of the SFRs in the package, and may also have one or more SFRs that are hierarchically higher than an SFR in the functional package.

CC Part 3 ACE_CCL.1.8C: *The conformance claim shall identify all assurance packages to which the PP-Module claims conformance.*

11.3.1.3.8 Work unit ACE_CCL.1-8

The evaluator ***shall check***, for each identified assurance package, that the package definition is complete. If the PP-Module does not claim conformance to an assurance package, this work unit is not applicable and therefore considered to be satisfied. If the assurance package is a reference to one of the assurance packages contained in CC Part 5 then this work unit is also considered to be satisfied. The evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 "Packages" by checking that the assurance package includes:

a) An assurance package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) An assurance package overview, giving a narrative description of the security functionality;

c) One or more security components or requirements (the assurance package SARs) drawn from CC Part 3, extended assurance components or some combination of both;

d) An assurance package shall not include an SPD or security objectives;

e) If extended components have been specified then the assurance package includes an extended components definition;

A security requirements rationale that provides the rationale for selecting the assurance components/requirements included in the package. CC Part 3 ACE_CCL.1.9C: *The conformance claim shall describe any conformance of the PP-Module to an assurance package as either package-conformant or package-augmented.*

11.3.1.3.9 Work unit ACE_CCL.1-9

The evaluator ***shall check*** that, for each identified assurance package, the conformance claim states a claim of either package-conformant or package-augmented.

If the PP-Module does not claim conformance to an assurance package, this work unit is not applicable and therefore considered to be satisfied.

If the assurance package conformance claim contains package-conformant, the evaluator determines that all constituent parts included in the assurance package are included in identical form by the PP-Module, without modification.

If the assurance package conformance claim contains package-augmented, the evaluator determines that all constituent parts of the assurance package included in the PP-Module are identical to those given in the assurance package except that the PP-Module shall contain at least one additional SAR or one SAR that is hierarchically higher than those contained in the assurance package.

CC Part 3 ACE_CCL.1.10C: *For exact conformance, the PP-Module's conformance statement shall contain an allowed-with statement that identifies the set of PPs and PP-Modules (exclusive of those PPs and PP-Modules that are included in the PP-Module Base) to which, in combination with the PP-Module under evaluation, exact conformance is allowed to be claimed.*

11.3.1.3.10 Work unit ACE_CCL.1-10

The evaluator ***shall check*** the conformance statement of the PP-Module to determine that it contains an allowed-with statement that lists the set of other PPs and PP-Modules that can be specified in the components statement of a PP-Configuration that includes the PP-Module.

The evaluator confirms that the listed PPs and PP-Modules are not part of any of the PP-Module Base sets.

There is no further check other than existence of the list for the PP-Module's allowed-with statement.

11.3.1.3.11 Work unit ACE_CCL.1-11

The evaluator shall check that, for each PP-Module Base for the PP-Module, all components in the PP-Module Base identify all other components of the PP-Module Base in their allow-with statements.

This is an iterative operation, and it is important to note that the rule about a PP-Module not including any components in its PP-Module Base in its allowed-with statement holds for any PP-Module in the PP-Module Base.

All PPs in a given PP-Module Base set will have to allow-with all other components (PPs and PP-Modules) in that PP-Module Base set.

PPs and PP-Modules in one PP-Module Base set do not, however, have to express an allowed with relationship with components in any other PP-Module Base set, since two PP-Module Base sets will not simultaneously be specified in a PP-Configuration

CC Part 3 ACE_CCL.1.11C: *The conformance statement may identify the set of CEM-derived Evaluation Methods and Evaluation Activities that shall be used with the PP-Module under evaluation. This list shall contain any Evaluation Methods and Evaluation Activities that are specified in the PP-Module but also any Evaluation Methods and Evaluation Activities specified in the PP-Module Base(s) and/or in the packages (if any) for which conformance is being claimed by the PP-Module under evaluation.*

11.3.1.3.12 Work unit ACE_CCL.1-12

The evaluator ***shall check*** the conformance statement of the PP-Module under evaluation to confirm that:

a) if any derived Evaluation Methods and Evaluation Activities are required by other items used with the PP-Module (e.g. a base PP), or required by other items to which the PP-Module claims conformance (e.g. packages), then these are all identified in the PP-Module under evaluation, along with any derived Evaluation Methods and Evaluation Activities that the PP-Module itself requires;

b) the list of derived Evaluation Methods and Evaluation Activities is sufficiently structured and detailed to unambiguously identify and locate every member of the list;

c) if there is any overlap in the scope of the identified Evaluation Methods and Evaluation Activities (i.e. where an overlay exists as described in CC Part 4) then the rationale for the resulting set of Evaluation Methods and Evaluation Activities is applicable to the TOE as described by the PP-Module under evaluation.

The intention of this work unit is to ensure that when evaluating a TOE that claims conformance with the PP-Configuration that contains the PP-Module under evaluation then the correct Evaluation Methods and Evaluation Activities can be used. This means that identification in the PP-Module need not list individual Evaluation Activities where these are unambiguously included in a listed Evaluation Method. Similarly, where multiple Evaluation Methods or Evaluation Activities are included in a single document then it is sufficient to reference the document, provided this does lead to unambiguous identification of the Evaluation Methods and Evaluation Activities that apply to the PP-Module under evaluation.

EXAMPLE If a document lists multiple different Evaluation Methods applicable to different use cases then it would not be sufficient to reference the document: the relevant use cases would also have to be identified.

11.4 PP-Module Security problem definition (ACE_SPD)

11.4.1 Evaluation of sub-activity (ACE_SPD.1)

11.4.1.1 Objectives

The objective of this sub-activity is to determine that the security problem intended to be addressed by the PP-Module and its operational environment is clearly defined.

11.4.1.2 Input

The evaluation evidence for this sub-activity is the PP-Module.

11.4.1.3 Action ACE_SPD.1.1E

11.4.1.3.1 General

CC Part 3 ACE_SPD.1.1C: *The security problem definition shall describe the threats.*

11.4.1.3.2 Work unit ACE_SPD.1-1

The evaluator ***shall check*** that the security problem definition describes the threats.

If all security objectives are derived from assumptions and/or OSPs only, the statement of threats need not be present in the PP-Module. In this case, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the security problem definition describes the threats that must be countered by the PP-Module and/or its operational environment.

Note that if optional requirements are defined by the PP-Module, there may be associated threats that are covered by this work unit.

CC Part 3 ACE_SPD.1.2C: *All threats shall be described in terms of a threat agent, an asset, and an adverse action.*

11.4.1.3.3 Work unit ACE_SPD.1-2

The evaluator ***shall examine*** the security problem definition to determine that all threats are described in terms of a threat agent, an asset, and an adverse action.

If all security objectives are derived from assumptions and OSPs only, the statement of threats need not be present in the PP-Module. In this case, this work unit is not applicable and therefore considered to be satisfied.

Threat agents may be further described by aspects such as expertise, resource, opportunity, and motivation.

CC Part 3 ACE_SPD.1.3C: *The security problem definition shall describe the OSPs.*

11.4.1.3.4 Work unit ACE_SPD.1-3

The evaluator ***shall examine*** that the security problem definition describes the OSPs.

If all security objectives are derived from assumptions and/or threats only, OSPs need not be present in the PP-Module. In this case, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that OSP statements are made in terms of rules or guidelines that must be followed by the TOE and/or its operational environment.

The evaluator determines that each OSP 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.

Note that if optional requirements are defined by the PP-Module, there may be associated OSPs that are covered by this work unit.

CC Part 3 ACE_SPD.1.4C: *The security problem definition shall describe the assumptions about the operational environment of the TOE.*

11.4.1.3.5 Work unit ACE_SPD.1-4

The evaluator ***shall examine*** the security problem definition to determine that it describes the assumptions about the operational environment of the TOE.

If there are no assumptions, this work unit is not applicable and is therefore considered to be satisfied.

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

11.5 PP-Module Security objectives (ACE_OBJ)

11.5.1 Evaluation of sub-activity (ACE_OBJ.1)

11.5.1.1 Application notes

If the PP-Configuration does not use the Direct Rationale approach (as determined in ACE_CCO.1-3) then this sub-activity can be considered satisfied.

11.5.1.2 Action ACE_OBJ.1.1E

11.5.1.2.1 General

CC Part 3 ACE_OBJ.1.1C: *The statement of security objectives shall describe the security objectives for the operational environment.*

11.5.1.2.2 Work unit ACE_OBJ.1-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the operational environment.

The evaluator checks that the security objectives for the operational environment are identified.

CC Part 3 ACE_OBJ.1.2C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

11.5.1.2.3 Work unit ACE_OBJ.1-2

The evaluator ***shall check*** that the security objectives requirements rationale traces the security objectives for the operational environment back to threats countered by that security objective, to OSPs enforced by that security objective, and to assumptions upheld by that security objective.

Each security objective for the operational environment may trace back to threats, OSPs, assumptions, or a combination of threats, OSPs and/or assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

There is no prescribed location where this tracing element of the rationale must be placed: for example, the relevant parts may be located under each threat, OSP and assumption in order to help make the security argument clearer and easier to read.

CC Part 3 ACE_OBJ.1.3C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

11.5.1.2.4 Work unit APE_OBJ.1-3

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that simply listing in the security objectives rationale the security objectives for the operational environment associated with each assumption may be a part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "Security Objective X directly upholds Assumption Y".

11.5.2 Evaluation of sub-activity (ACE_OBJ.2)

11.5.2.1 Objectives

The objective of this sub-activity is to determine whether the security objectives adequately and completely address the security problem definition and that the division of this problem between the TOE and its operational environment is clearly defined.

11.5.2.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Module.

11.5.2.3 Action ACE_OBJ.2.1E

11.5.2.3.1 General

CC Part 3 ACE_OBJ.2.1C: *The statement of security objectives shall describe the security objectives for the TOE and the security objectives for the operational environment.*

11.5.2.3.2 Work unit ACE_OBJ.2-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the TOE and the security objectives for the operational environment.

The evaluator checks that both categories of security objectives are clearly identified and separated from the other category.

CC Part 3 ACE_OBJ.2.2C: *The security objectives rationale shall trace each security objective for the TOE back to threats countered by that security objective and OSPs enforced by that security objective.*

11.5.2.3.3 Work unit ACE_OBJ.2-2

The evaluator ***shall check*** that the security objectives rationale traces all security objectives for the TOE back to threats countered by the objectives and/or OSPs enforced by the objectives.

Each security objective for the TOE may trace back to threats or OSPs, or a combination of threats and OSPs, but it must trace back to at least one threat or OSP. Optional requirements may require Threats/OSPs to be specified, and security objectives associated with these SPD elements are also covered by this work unit.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the TOE has no useful purpose.

CC Part 3 ACE_OBJ.2.3C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

11.5.2.3.4 Work unit ACE_OBJ.2-3

The evaluator ***shall check*** that the security objectives rationale traces the security objectives for the operational environment back to threats countered by that security objective, to OSPs enforced by that security objective, and to assumptions upheld by that security objective.

Each security objective for the operational environment may trace back to threats, OSPs, assumptions, or a combination of threats, OSPs and/or assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

CC Part 3 ACE_OBJ.2.4C: *The security objectives rationale shall demonstrate that the security objectives counter all threats.*

11.5.2.3.5 Work unit ACE_OBJ.2-4

The evaluator ***shall examine*** the security objectives rationale to determine that it justifies for each threat that the security objectives are suitable to counter that threat.

If no security objectives trace back to the threat, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

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

Note that the tracings from security objectives to threats provided in the security objectives rationale may be 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 may be as minimal as "Security Objective X directly counters Threat Y".

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

CC Part 3 ACE_OBJ.2.5C: *The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.*

The evaluator ***shall examine*** the security objectives rationale to determine that for each OSP it justifies that the security objectives are suitable to enforce that OSP.

If no security objectives trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all security objectives that trace back to that OSP are achieved, the OSP is enforced.

The evaluator also determines that each security objective that traces back to an OSP is necessary: when the security objective is achieved it actually contributes to the enforcement of the OSP.

Note that the tracings from security objectives to OSPs provided in the security objectives rationale may be part of a justification, but do not constitute a justification by themselves. In the case that a security objective is merely a statement reflecting the intent to enforce a particular OSP, a justification is required, but this justification may be as minimal as "Security Objective X directly enforces OSP Y".

CC Part 3 ACE_OBJ.2.6C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

11.5.2.3.6 Work unit ACE_OBJ.2-6

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that the tracings from security objectives for the operational 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 operational environment is merely a restatement of an assumption, a justification is required, but this justification may be as minimal as "Security Objective X directly upholds Assumption Y".

11.6 PP-Module extended components definition (ACE_ECD)

11.6.1 Evaluation of sub-activity (ACE_ECD.1)

11.6.1.1 Objectives

The objective of this sub-activity is to determine whether extended components have been clearly and unambiguously defined, and whether they are necessary, i.e. they may not be clearly expressed using existing CC Part 2 or CC Part 3 components.

11.6.1.2 Input

The evaluation evidence for this sub-activity is the PP-Module.

11.6.1.3 Action ACE_ECD.1.1E

11.6.1.3.1 General

CC Part 3 ACE_ECD.1.1C: *The statement of security requirements shall identify all the extended security requirements.*

11.6.1.3.2 Work unit ACE_ECD.1-1

The evaluator ***shall check*** that all security requirements in the statement of security requirements that are not identified as extended requirements are present in CC Part 2 or in CC Part 3.

CC Part 3 ACE_ECD.1.2C: *The extended components definition shall define an extended component for each extended security requirement.*

11.6.1.3.3 Work unit ACE_ECD.1-2

The evaluator ***shall check*** that the extended components definition defines an extended component for each extended security requirement.

If the PP-Module does not contain extended security requirements, this work unit is not applicable and therefore considered to be satisfied.

A single extended component may be used to define multiple iterations of an extended security requirement; it is not necessary to repeat this definition for each iteration.

CC Part 3 ACE_ECD.1.3C: *The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.*

11.6.1.3.4 Work unit ACE_ECD.1-3APE_ECD.1-3

The evaluator ***shall examine*** the extended components definition to determine that it describes how each extended component fits into the existing CC components, families, and classes.

If the PP-Module does not contain extended security requirements, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that each extended component is either:

a) a member of an existing CC Part 2 or CC Part 3 family; or

b) a member of a new family defined in the PP-Module.

If the extended component is a member of an existing CC Part 2 or CC Part 3 family, the evaluator determines that the extended components definition adequately describes why the extended component should be a member of that family and how it relates to other components of that family.

If the extended component is a member of a new family defined in the PP-Module, the evaluator confirms that the extended component is not appropriate for an existing family.

If the PP-Module defines new families, the evaluator determines that each new family is either:

a) a member of an existing CC Part 2 or CC Part 3 class; or

b) a member of a new class defined in the PP-Module.

If the family is a member of an existing CC Part 2 or CC Part 3 class, the evaluator determines that the extended components definition adequately describes why the family should be a member of that class and how it relates to other families in that class.

If the family is a member of a new class defined in the PP-Module, the evaluator confirms that the family is not appropriate for an existing class.

11.6.1.3.5 Work unit ACE_ECD.1-4APE_ECD.1-4

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended component identifies all applicable dependencies of that component.

If the PP-Module does not contain extended security requirements, this work unit is not applicable and therefore considered to be satisfied.

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

CC Part 3 ACE_ECD.1.4C: *The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.*

11.6.1.3.6 Work unit ACE_ECD.1-5

The evaluator ***shall examine*** the extended components definition to determine that each extended functional component uses the existing CC Part 2 components as a model for presentation.

If the PP-Module does not contain extended SFRs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended functional component is consistent with CC Part 2, 6.1.3, Component structure.

If the extended functional component uses operations, the evaluator determines that the extended functional component is consistent with CC Part 1, 8.2. Operations.

If the extended functional component is hierarchical to an existing functional component, the evaluator determines that the extended functional component is consistent with CC Part 2, 6.2.1, Component changes highlighting.

11.6.1.3.7 Work unit ACE_ECD.1-6

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional family uses the existing CC functional families as a model for presentation.

If the PP-Module does not define new functional families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional families are defined consistent with CC Part 2, 6.1.2, Family structure.

11.6.1.3.8 Work unit ACE_ECD.1-7

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional class uses the existing CC functional classes as a model for presentation.

If the PP-Module does not define new functional classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional classes are defined consistent with CC Part 2, 6.1.1, Class structure

11.6.1.3.9 Work unit ACE_ECD.1-8

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended assurance component uses the existing CC Part 3 components as a model for presentation.

If the PP-Module does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended assurance component definition is consistent with CC Part 3, 6.1.3, Assurance component structure.

If the extended assurance component uses operations, the evaluator determines that the extended assurance component is consistent with CC Part 1, 8.2, Operations.

If the extended assurance component is hierarchical to an existing assurance component, the evaluator determines that the extended assurance component is consistent with CC Part 3, 6.1.3, Assurance component structure.

11.6.1.3.10 Work unit ACE_ECD.1-9

The evaluator ***shall examine*** the extended components definition to determine that, for each defined extended assurance component, applicable methodology has been provided.

If the PP-Module does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that, for each evaluator action element of each extended SAR, one or more work units are provided and that successfully performing all work units for a given evaluator action element will demonstrate that the element has been achieved.

11.6.1.3.11 Work unit ACE_ECD.1-10

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance family uses the existing CC assurance families as a model for presentation.

If the PP-Module does not define new assurance families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance families are defined consistent with CC Part 3, 6.1.2, Assurance family structure.

11.6.1.3.12 Work unit ACE_ECD.1-11

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance class uses the existing CC assurance classes as a model for presentation.

If the PP-Module does not define new assurance classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance classes are defined consistent with CC Part 3, 6.1.1, Assurance class structure.

CC Part 3 ACE_ECD.1.5C: *The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.*

11.6.1.3.13 Work unit ACE_ECD.1-12

The evaluator ***shall examine*** the extended components definition to determine that each element in each extended component is measurable and states objective evaluation requirements, such that conformance or nonconformance can be demonstrated.

If the PP-Module does not contain extended security requirements, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that elements of extended functional components are stated in such a way that they are testable, and traceable through the appropriate TSF representations.

The evaluator also determines that elements of extended assurance components avoid the need for subjective evaluator judgement.

The evaluator is reminded that whilst being measurable and objective is appropriate for all evaluation criteria, it is acknowledged that no formal method exists to prove such properties. Therefore, the existing CC functional and assurance components are to be used as a model for determining what constitutes conformance to this requirement.

11.6.1.4 Action ACE_ECD.1.2E

11.6.1.4.1 Work unit ACE_ECD.1-13

The evaluator ***shall examine*** the extended components definition to determine that each extended component may not be clearly expressed using existing components.

If the PP-Module does not contain extended security requirements, this work unit is not applicable and therefore considered to be satisfied.

The evaluator should take components from CC Part 2 and CC Part 3, other extended components that have been defined in the PP-Module, combinations of these components, and possible operations on these components into account when making this determination.

The evaluator is reminded that the role of this work unit is to preclude unnecessary duplication of components, that is, components that may be clearly expressed by using other components. The evaluator should not undertake an exhaustive search of all possible combinations of components including operations in an attempt to find a way to express the extended component by using existing components.

11.7 PP-Module security requirements (ACE_REQ)

11.7.1 Evaluation of sub-activity (ACE_REQ.1)

11.7.1.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs counter the threats and implement the organisational security policies of the TOE.

11.7.1.2 Input

The evaluation evidence for this sub-activity is the PP-Module.

11.7.1.3 Action ACE_REQ.1.1E

11.7.1.3.1 General

CC Part 3 ACE_REQ.1.1C: *The statement of security requirements shall describe the SFRs and SARs (the SARs that apply to the PP-Module may be explicitly stated, or inherited from the PP-Module Base(s)).*

11.7.1.3.2 Work unit ACE_REQ.1-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2;

b) by reference to an extended component in the extended components definition of the PP-Module Base;

c) by reference to a security requirements package that the PP-Module claims to be conformant with;

d) by reproduction in the PP-Module.

It is not required to use the same means of identification for all SFRs.

11.7.1.3.3 Work unit ACE_REQ.1-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that each SAR is identified by one of the following means:

a) by reference to an individual component in CC Part 3;

b) by reference to an extended component in the extended components definition of the PP-Module Base;

c) by reference to a security requirements package that the PP-Module claims to be conformant with;

d) by reproduction in the PP-Module.

It is not required to use the same means of identification for all SARs.

CC Part 3 ACE_REQ.1.2C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

11.7.1.3.4 Work unit ACE_REQ.1-3

The evaluator ***shall examine*** the PP-Module to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the PP-Module defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the PP-Module author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This can be especially applicable if the same terms are used in the rest of the PP-Module.

CC Part 3 ACE_REQ.1.3C: *The statement of security requirements shall identify all operations on the security requirements.*

11.7.1.3.5 Work unit ACE_REQ.1-4

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

The evaluator determines that all operations are identified in each SFR or SAR where such an operation is used. This includes both completed operations and uncompleted operations. Identification may be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

If the PP-Module defines *selection-based* SFRs, the evaluator determines that the PP-Module clearly identifies the dependencies between the selection in an SFR and the selection-based SFR(s) to be included in the ST should that selection be chosen by the ST author.

CC Part 3 ACE_REQ.1.4C: *All operations shall be performed correctly.*

11.7.1.3.6 Work unit ACE_REQ.1-5

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.1.3.7 Work unit ACE_REQ.1-6

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.1.3.8 Work unit ACE_REQ.1-7

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.1.3.9 Work unit ACE_REQ.1-8

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

CC Part 3 ACE_REQ.1.5C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

11.7.1.3.10 Work unit ACE_REQ.1-9

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that the security requirements rationale justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

- why the dependency is not necessary or useful, in which case no further information is required; or

- that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

The evaluator ensures that if a functional package defines a dependency on another functional package, then that functional package is included in the PP-Module.

If a functional package identifies dependencies on its requirements that need to be satisfied by the underlying PP-Module, the evaluator ensures that their analysis covers these dependencies as well.

CC Part 3 ACE_REQ.1.6C: *The security requirements rationale shall trace each SFR back to the threats countered by that SFR and the OSPs enforced by that SFR.*

11.7.1.3.11 Work unit ACE_REQ.1-10

The evaluator ***shall check*** that the security requirements rationale traces each SFR back to the threats countered by that SFR and OSPs enforced by that SFR.

The evaluator determines that each SFR is traced back to at least one threat or OSP for the TOE.

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

There is no prescribed location where this tracing element of the rationale must be placed: for example, the relevant parts may be located under each threat and OSP in order to help make the security argument clearer and easier to read.

CC Part 3 ACE_REQ.1.7C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) counter all the threats for the TOE.*

11.7.1.3.12 Work unit ACE_REQ.1-11

The evaluator ***shall examine*** the security requirements rationale to determine that for each threat it demonstrates that the SFRs are suitable to meet that threat.

If no SFRs trace back to a threat, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

The evaluator determines that the justification for a threat demonstrates that the SFRs are sufficient: if all SFRs that trace back to the threat are achieved then, in the context of any applicable OSPs and assumptions, the threat is removed, sufficiently diminished, or the effects of the threat are sufficiently mitigated.

Note that simply listing in the security requirements rationale the SFRs associated with each threat may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly counters Threat Y".

The evaluator also determines that each SFR that traces back to a threat is necessary: when the SFR is implemented it actually contributes to the removal, diminishing or mitigation of that threat.

CC Part 3 ACE_REQ.1.8C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) enforce all the OSPs for the TOE.*

11.7.1.3.13 Work unit ACE_REQ.1-12

The evaluator ***shall examine*** the security requirements rationale to determine that for each OSP it justifies that the SFRs are suitable to enforce that OSP.

If no SFRs or security objectives for the operational environment trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all SFRs that trace back to that OSP are achieved then, in the context of any applicable assumptions, the OSP is enforced.

The evaluator also determines that each SFR that traces back to an OSP is necessary: when the SFR is implemented it actually contributes to the enforcement of the OSP.

Note that simply listing in the security requirements rationale the SFRs associated with each OSP may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly enforces OSP Y".

CC Part 3 ACE_REQ.1.9C: *The security requirements rationale shall explain why the SARs were chosen.*

11.7.1.3.14 Work unit ACE_REQ.1-13

The evaluator ***shall check*** that the security requirements rationale explains why the SARs were chosen.

The evaluator is reminded that any explanation is correct, as long as it is coherent and neither the SARs nor the explanation have obvious inconsistencies with the remainder of the PP-Module.

An example of an obvious inconsistency between the SARs and the remainder of the PP-Module would be to have threat agents that are very capable, but an AVA_VAN component that does not protect against these threat agents.

CC Part 3 ACE_REQ.1.10C: *The statement of security requirements shall be internally consistent.*

11.7.1.3.15 Work unit ACE_REQ.1-14

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

- all optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

- all optional requirements are clearly identified as either:

    1) elective and therefore for inclusion purely at the discretion of the ST author (i.e. conformance to the PP-Module does not depend on their inclusion in the ST); or

    2) conditional and therefore required if a conformant TOE implements the functionality covered by the requirement.

- all optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to "all objects", "all subjects" etc., that these requirements do not conflict.

Some possible conflicts are:

- an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended SAR specifying an open source review;

- FAU_Gen.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

- FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

- multiple iterations of FDP_ACC.1 Subset access control, especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

11.7.2 Evaluation of sub-activity (ACE_REQ.2)

11.7.2.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs meet the security objectives of the TOE.

11.7.2.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Module.

11.7.2.3 Action ACE_REQ.2.1E

11.7.2.3.1 General

CC Part 3 ACE_REQ.2.1C: *The statement of security requirements shall describe the SFRs and SARs (the SARs that apply to the PP-Module may be explicitly stated, or inherited from the PP-Module Base(s)).*

11.7.2.3.2 Work unit ACE_REQ.2-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2

b) by reference to an extended component in the extended components definition of the PP-Module Base;

c) by reference to an individual component in a security requirements package that the PP-Module claims to be conformant with;

d) by reproduction in the PP-Module.

It is not required to use the same means of identification for all SFRs.

11.7.2.3.3 Work unit ACE_REQ.2-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that each SAR is identified by one of the following means:

a) by reference to an individual component in CC Part 3;

b) by reference to an extended component in the extended components definition of the PP-Module Base;

c) by reference to an individual component in a security requirements package that the PP-Module claims to be conformant with;

d) by reproduction in the PP-Module.

It is not required to use the same means of identification for all SARs.

Note that if optional requirements are defined by the PP-Module, there may be associated threats that are covered by this work unit.

CC Part 3 ACE_REQ.2.2C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

11.7.2.3.4 Work unit ACE_REQ.2-3

The evaluator ***shall examine*** the PP-Module to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the PP-Module defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the PP-Module author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This may be especially applicable if the same terms are used in the rest of the PP-Module.

CC Part 3 ACE_REQ.2.3C: *The statement of security requirements shall identify all operations on the security requirements.*

11.7.2.3.5 Work unit ACE_REQ.2-4

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

The evaluator determines that all operations are identified in each SFR or SAR where such an operation is used. This includes both completed operations and uncompleted operations. Identification may be achieved by typographical distinctions, or by explicit identification in the surrounding text, or by any other distinctive means.

If the PP-Module defines *selection-based* SFRs, the evaluator determines that the PP-Module clearly identifies the dependencies between the selection in an SFR and the selection-based SFR(s) to be included in the ST should that selection be chosen by the ST author.

CC Part 3 ACE_REQ.2.4C: *All operations shall be performed correctly.*

11.7.2.3.6 Work unit ACE_REQ.2-5

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.2.3.7 Work unit ACE_REQ.2-6

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.2.3.8 Work unit ACE_REQ.2-7

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

11.7.2.3.9 Work unit ACE_REQ.2-8

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2, Operations.

CC Part 3 ACE_REQ.2.5C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

11.7.2.3.10 Work unit ACE_REQ.2-9

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that the security requirements rationale justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

a) why the dependency is not necessary or useful, in which case no further information is required; or

b) that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

The evaluator ensures that if a functional package defines a dependency on another functional package, then that functional package is included in the PP-Module.

If a functional package identifies dependencies on its requirements that need to be satisfied by the underlying PP-Module, the evaluator ensures that their analysis covers these dependencies as well.

CC Part 3 ACE_REQ.2.6C: *The security requirements rationale shall trace each SFR back to the security objectives for the TOE enforced by that SFR.*

11.7.2.3.11 Work unit ACE_REQ.2-10

The evaluator ***shall check*** that the security requirements rationale traces each SFR back to the security objectives for the TOE.

Optional requirements may require Threats/OSPs to be specified, and security objectives associated with these SPD elements are also covered by this work unit.

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

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

CC Part 3 ACE_REQ.2.7C: *The security requirements rationale shall demonstrate that the SFRs meet all security objectives for the TOE.*

11.7.2.3.12 Work unit ACE_REQ.2-11

The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the TOE it justifies that the SFRs are suitable to meet that security objective for the TOE.

If no SFRs trace back to the security objective for the TOE, the evaluator action related to this work unit is assigned a fail verdict.

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

If the SFRs that trace back to a security objective for the TOE have any uncompleted assignments, or uncompleted or restricted selections, the evaluator determines that for every conceivable completion or combination of completions of these operations, the security objective is still met.

The evaluator also determines that each SFR that traces back to a security objective for the TOE is necessary: when the SFR is satisfied, it actually contributes to achieving the security objective.

Note that the tracings from SFRs 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.

CC Part 3 ACE_REQ.2.8C: *The security requirements rationale shall explain why the SARs were chosen.*

11.7.2.3.13 Work unit ACE_REQ.2-12

The evaluator ***shall check*** that the security requirements rationale explains why the SARs were chosen.

The evaluator is reminded that any explanation is correct, as long as it is coherent and neither the SARs nor the explanation have obvious inconsistencies with the remainder of the PP-Module.

An example of an obvious inconsistency between the SARs and the remainder of the PP-Module would be to have threat agents that are very capable, but an AVA_VAN component that does not protect against these threat agents.

CC Part 3 ACE_REQ.2.9C: *The statement of security requirements shall be internally consistent.*

11.7.2.3.14 Work unit ACE_REQ.2-13

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

- All optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

- All optional requirements are clearly identified as either:

1) elective and therefore for inclusion purely at the discretion of the ST author (i.e. conformance to the PP-Module does not depend on their inclusion in the ST); or

2) conditional and therefore required if a conformant TOE implements the functionality covered by the requirement.

- All optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to "all objects", "all subjects" etc., that these requirements do not conflict.

Some possible conflicts are:

- an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended SAR specifying an open source review;

- FAU_GEN.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

- FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

- multiple iterations of FDP_ACC.1 Subset access control, especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

11.8 PP-Module consistency (ACE_MCO)

11.8.1 Evaluation of sub-activity (ACE_MCO.1)

11.8.1.1 Objectives

The objective of this sub-activity is to determine the consistency of the PP-Module regarding its PP-Module Base.

11.8.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Module;

b) its PP-Module Base.

11.8.1.3 Action ACE_MCO.1.1E

11.8.1.3.1 General

CC Part 3 ACE_MCO.1.1C: *The consistency rationale shall demonstrate that the TOE type of the PP-Module and the TOE types of its PP-Module Base(s) are consistent.*

11.8.1.3.2 Work unit ACE_MCO.1-1

The evaluator ***shall examine*** the consistency rationale to determine that the TOE type of the PP-Module is consistent with all the TOE types of the *PP-Module Base(s)*.

The relation between the types may be simple: a PP-Module may consider a TOE that provides additional security functionality, or more complex: a TOE that provides a given security functionality in a specific way.

CC Part 3 ACE_MCO.1.2C: *The consistency rationale shall identify the assets of the PP-Module's SPD that also belong to some of its PP-Module Bases and amongst them those for which the PP-Module and the PP-Module Base define different security problems.*

11.8.1.3.3 Work unit ACE_MCO.1-2

The evaluator ***shall check*** that the consistency rationale contains the set of assets shared between the PP-Module and PP-Module Base(s), and that this set is unambiguous and complete.

The evaluator ***shall check*** that the consistency rationale contains the subset of shared assets that hold different security properties and/or are subject to different threat agents or threat scenarios, and that this subset is unambiguous and complete.

If there is no shared asset, then this work unit is considered satisfied.

CC Part 3 ACE_MCO.1.3C: *The consistency rationale shall demonstrate that:*

- *the statement of the security problem definition is consistent with the statement of the security problem definition of its PP-Module Base(s);*
- *the statement of the security problem definition is consistent with the statement of the security problem definition of any functional package for which conformance is being claimed.*

11.8.1.3.4 Work unit ACE_MCO.1-3

The evaluator ***shall examine*** the PP-Module consistency rationale to determine that it demonstrates that the statement of security problem definition of the PP-Module is consistent with the statements of security problem definition stated in its PP-Module Base(s).

In particular, the evaluator examines the consistency rationale to determine that:

a) the statements of threats, assumptions and OSPs in the PP-Module do not contradict those drawn from any functional packages to which it claims conformance;

b) the statement of assumptions in the PP-Module addresses aspects out of scope of any functional packages to which it claims conformance, in which case, the addition of elements is allowed.

11.8.1.3.5 Work unit ACE_MCO.1-4

The evaluator ***shall examine*** the PP-Module consistency rationale to determine that it demonstrates that the statement of security problem definition of the PP-Module is consistent

with the statements of security problem definition stated in any of the functional packages for which conformance is being claimed.

In particular, the evaluator examines the consistency rationale to determine that:

a) the statements of threats, assumptions and OSPs in the PP-Module do not contradict those from the PP-Module Base;

b) the statement of assumptions in the PP-Module addresses aspects out of scope of the PP-Module Base, in which case, the addition of elements is allowed.

11.8.1.3.6 Work unit ACE_MCO.1-5

For all the assets that are shared between the PP-Module and a base PP or PP-Module, the evaluator ***shall determine*** that all the differences in the security problem definitions are justified. For instance, the asset resides in different locations or at different times or is subject to different operational environment conditions.

In particular, the evaluator examines the consistency rationale to determine that:

a) the statements of security objectives for the TOE and the security objectives for the operational environment the PP-Module do not contradict those drawn from any functional packages to which it claims conformance;

b) the statement of security objectives for the operational environment in the PP-Module addresses aspects out of scope of any functional packages to which it claims conformance, in which case, the addition of elements is allowed.

CC Part 3 ACE_MCO.1.4C: *The consistency rationale shall demonstrate that:*

- *the security objectives definition is consistent with the security objectives of its PP-Module Base(s);*

- *the security objectives definition is consistent with the security objectives of any functional package for which conformance is being claimed*.

11.8.1.3.7 Work unit ACE_MCO.1-6

The evaluator ***shall examine*** the PP-Module consistency rationale to determine that it demonstrates that the statement of security objectives of the PP-Module is consistent with the statement of security objectives of its PP-Module Base.

11.8.1.3.8 Work unit ACE_MCO.1-7

The evaluator ***shall examine*** the PP-Module consistency rationale to determine that it demonstrates that the statement of security objectives of the PP-Module is consistent with the statement of security objectives of any functional package for which conformance is being claimed.

Where the PP-Module and its PP-Module Base use the Direct Rationale approach then this work unit is trivially satisfied for the TOE objectives (because these are not included under the Direct Rationale approach). If *any* of the PP-Module or its PP-Module Base use the Direct Rationale approach then the PP-Module *and all* elements of its PP-Module Base must use the Direct Rationale approach, otherwise the evaluator action related to this work unit is assigned a fail verdict.

In particular, the evaluator examines the consistency rationale to determine that:

a) the statements of the security objectives for the TOE and the security objectives for the operational environment in the PP-Module do not contradict those from the PP-Module Base;

b) the statement of the security objectives for the operational environment in the PP-Module addresses aspects out of scope of the PP-Module Base, in which case, the addition of elements is allowed.

CC Part 3 ACE_MCO.1.5C: *The consistency rationale shall demonstrate that:*

- *the security functional requirements definition is consistent with the security functional requirements of its PP-Module Base(s);*

- *the security functional requirements definition is consistent with the security functional requirements of any functional package for which conformance is being claimed.*

11.8.1.3.9 Work unit ACE_MCO.1-8

The evaluator ***shall examine*** the consistency rationale to determine that the statement of security requirements of the PP-Module is consistent with the statement of security requirements of its PP-Module Base, i.e. the SFRs of the PP-Module either complete or refine the SFRs of the PP-Module Base and no contradiction arises from the whole set of SFRs of the PP-Module and the PP-Module Base.

11.8.1.3.10 Work unit ACE_MCO.1-9

The evaluator ***shall examine*** the consistency rationale to determine that the statement of security requirements of the PP-Module is consistent with the statement of security requirements of any functional package for which conformance is being claimed, i.e. the SFRs of the PP-Module either complete or refine the SFRs of the claimed functional packages and no contradiction arises from the whole set of SFRs of the PP-Module and those of the functional packages for which conformance is claimed.

CC Part 3 ACE_MCO.1.6C: *The assurance rationale shall demonstrate the internal consistency of the set of security assurance requirements of the PP-Module with regard to its security problem definition.*

11.8.1.3.11 Work unit ACE_MCO.1-10

The evaluator ***shall examine*** the assurance rationale to determine that the set of security assurance requirements of the PP-Module is consistent with regard to the threat model defined in the PP-Module.

An example of an inconsistency between SARs and the SPD would be to consider highly skilled threat agents together with a low AVA_VAN level that cannot consider these threat agents by definition.

CC Part 3 ACE_MCO.1.7C: *The assurance rationale shall demonstrate the consistency of the set of security assurance requirements of the PP-Module with regard to the security assurance requirements of the PP-Module Base(s).*

11.8.1.3.12 Work unit ACE_MCO.1-11

The evaluator ***shall examine*** the assurance rationale to determine that the set of security assurance requirements of the PP-Module is consistent with regard to all the sets of SARs defined in its PP-Module Base.

11.8.1.3.13 Work unit ACE_MCO.1-12

The evaluator ***shall examine*** the assurance rationale to determine that the set of SARs defined in the PP-Module does not undermine the security that is expected for the assets that are shared between the PP-Module and its PP-Module Base (if shared assets exist).

11.9 PP-Configuration consistency (ACE_CCO)

11.9.1 Evaluation of sub-activity (ACE_CCO.1)

11.9.1.1 Objectives

The objective of this sub-activity is to determine whether the PP-Configuration and its components are correctly identified.

The objective of this sub-activity is also to determine the consistency of the PP-Configuration regarding the whole set of PP and PP-Modules.

For the consistency analysis required by this activity, the application notes of the CEM, Section 10.8 (Re-using the evaluation results of certified PPs), is applicable to determine which parts of the PP-Module Base are to be re-evaluated during the evaluation of PP-Configuration.

11.9.1.2 Input

The evaluation evidence for this sub-activity is:

a) the PP-Configuration;

b) the PPs and PP-Modules identified in the components statement.

11.9.1.3 Action ACE_CCO.1.1E

11.9.1.3.1 General

CC Part 3 ACE_CCO.1.1C: *The PP-Configuration reference shall uniquely identify the PP-Configuration.*

11.9.1.3.2 Work unit ACE_CCO.1-1

The evaluator ***shall examine*** the PP-Configuration reference to determine that it uniquely identifies the PP-Configuration.

The evaluator determines that the PP-Configuration reference identifies the PP-Configuration itself, so that it may be easily distinguished from other PPs, PP-Configurations and PP-Modules, and that it also uniquely identifies each version of the PP-Configuration, e.g. by including a version number and/or a date of publication.

The PP-Configuration should have some referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates).

CC Part 3 ACE_CCO.1.2C: *The PP-Configuration components statement shall uniquely identify the PPs and PP-Modules that compose the PP-Configuration.*

11.9.1.3.3 Work unit ACE_CCO.1-2

The evaluator ***shall examine*** the PP-Configuration components statement to determine that it uniquely identifies the PPs and PP-Modules contained in the PP-Configuration.

11.9.1.3.4 Work unit ACE_CCO.1-3

The evaluator ***shall check*** that if *any* of the PPs/PP-Modules in the PP-Configuration use the Direct Rationale Approach then *all* the PPs and PP-Modules in the PP-Configuration use the Direct Rationale approach.

11.9.1.3.5 Work unit ACE_CCO.1-4

The evaluator ***shall check*** that if *any* of the PPs/PP-Modules in the PP-Configuration are of exact conformance type then *all* PPs/PP-Modules in the PP-Configuration are of exact conformance type.

The PPs should have been certified and available for use in Security Targets.

CC Part 3 ACE_CCO.1.3C: *For each PP-Module identified in the PP-Configuration components statement, the components statement shall include the PP-Module Base required by the identified PP-Module. If the PP-Module specifies alternative PP-Module Bases, only one of these PP-Module Bases shall be referred to in the PP-Configuration.*

11.9.1.3.6 Work unit ACE_CCO.1-5

The evaluator ***shall check*** that the PP-Module Base of each PP-Module in the PP-Configuration is identified in the PP-Configuration's component statement. Where a PP-Module specifies alternative PP-Module Bases then only one of these PP-Module Bases must be referred to in the PP-Configuration.

CC Part 3 ACE_CCO.1.4C: *For a multi-assurance PP-Configuration, the components statement shall describe the organisation of the TSF in terms of the sub-TSFs defined in the PPs and PP-Modules defined in the PP-Configuration.*

11.9.1.3.7 Work unit ACE_CCO.1-6

The evaluator ***shall check*** that the components statement provides a description of the TSF organisation in terms of the sub-TSFs defined by the PP-Configuration components.

CC Part 3 ACE_CCO.1.5C: *The TOE overview shall identify the TOE type.*

11.9.1.3.8 Work unit ACE_CCO.1-7

The evaluator ***shall check*** that the TOE overview identifies the TOE type.

CC Part 3 ACE_CCO.1.6C: *The TOE overview shall describe the usage and major security features of the TOE.*

11.9.1.3.9 Work unit ACE_CCO.1-8

The evaluator ***shall examine*** the TOE overview to determine that it describes the usage and major security features of the TOE.

The TOE overview should briefly (i.e. several paragraphs) describe the usage and major security features expected of the TOE. The TOE overview should enable consumers and potential TOE developers to quickly determine whether the PP-Configuration is of interest to them.

The evaluator determines that the overview is clear enough for TOE developers and consumers, and sufficient to give them a general understanding of the intended usage and major security features of the TOE.

CC Part 3 ACE_CCO.1.7C: *The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.*

11.9.1.3.10 Work unit ACE_CCO.1-9

The evaluator ***shall examine*** the TOE overview to determine that it identifies any non-TOE hardware/software/firmware available to the TOE.

While some TOEs may run stand-alone, other TOEs (notably software TOEs) need additional hardware, software or firmware to operate. In this subclause of the PP-Configuration, the PP-Configuration author lists all hardware, software, and/or firmware that will be available for the TOE to run on.

This identification should be detailed enough for potential consumers and TOE developers to determine whether their TOE may operate with the listed hardware, software and firmware.

CC Part 3 ACE_CCO.1.8C: *The conformance claim shall identify the CC edition(s) to which the PP-Configuration components claim conformance.*

11.9.1.3.11 Work unit ACE_CCO.1-10

The evaluator ***shall check*** that the conformance claim identifies the CC edition(s) to which the PP-Configuration and its components claim conformance.

CC Part 3 ACE_CCO.1.9C: *The conformance claim shall describe the conformance of the PP-Configuration to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.*

11.9.1.3.12 Work unit ACE_CCO.1-11

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 2 conformant or CC Part 2 extended for the PP-Configuration.

CC Part 3 ACE_CCO.1.10C: *The conformance claim shall describe the conformance of the PP-Configuration to this document as either "CC Part 3 conformant" or CC Part 3 extended".*

11.9.1.3.13 Work unit ACE_CCO.1-12

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 3 conformant or CC Part 3 extended for the PP-Configuration.

CC Part 3 ACE_CCO.1.11C: *The conformance claim shall be consistent with the conformance claims of the PP-Configuration components.*

11.9.1.3.14 Work unit ACE_CCO.1-13

The evaluator ***shall examine*** the PP-Configuration conformance claim to determine the compatibility between all CC versions that are related to the PP-Configuration and its components.

CC versions used in a PP-Configuration and its components have to be compatible. If compatibility is not obvious, guidance from the certification scheme should be asked.

CC Part 3 ACE_CCO.1.12C: *The conformance claim of a PP-Configuration shall include an assurance package conformance claim consisting of statements describing any conformance of the PP-Configuration to an assurance package as either package-conformant or package-augmented.*

11.9.1.3.15 Work unit ACE_CCO.1-14

The evaluator ***shall check*** that, for each identified assurance package, the conformance claim states a claim of either package-conformant or package-augmented.

If the PP-Configuration does not claim conformance to an assurance package, this work unit is not applicable and therefore considered to be satisfied.

If the assurance package conformance claim contains package-conformant, the evaluator determines that all constituent parts included in the assurance package are included in identical form by the PP-Configuration, without modification.

If the assurance package conformance claim contains package-augmented, the evaluator determines that all constituent parts of the assurance package included in the PP-Configuration are identical to those given in the assurance package except that the PP-Configuration shall contain at least one additional SAR or one SAR that is hierarchically higher than those contained in the assurance package.

CC Part 3 ACE_CCO.1.13C: *The conformance statement shall specify the required conformance to the PP-Configuration as one of exact, strict, demonstrable, or it shall provide the list of conformance types that are required by each of the PP-Configuration components.*

11.9.1.3.16 Work unit ACE_CCO.1-15

The evaluator ***shall examine*** the PP-Configuration conformance statement to determine that it specifies the kind of conformance required: exact, strict, demonstrable, or a list of strict and demonstrable.

If at least one of the PPs identified in the PP-Configuration components statement requires exact conformance, then the PP-Configuration conformance statement shall also require exact conformance.

CC Part 3 ACE_CCO.1.14C: *For the exact conformance case, the allowed-with statement of the conformance statement of each PP included in the components statement of the PP-Configuration shall identify all the PP-Configuration components as being allowed to be used in combination with the PP in a PP-Configuration.*

11.9.1.3.17 Work unit ACE_CCO.1-16

For the exact conformance case, for each PP listed in the PP-Configuration's components statement, the evaluator **shall check** the PP's conformance statement to determine that all PPs and PP-Modules specified in the PP-Configuration's components statement are listed as allowed to be used with that PP in a PP-Configuration.

For the exact conformance case, through ACE_CCO.1-16 and ACE_CCO.1-17 together, the evaluator ensures that any pair of components listed in the PP-Configuration components statement allow to use any other through allow with statement in components' conformance statement or PP-module base identification.

CC Part 3 ACE_CCO.1.15C: *For the exact conformance case, the allowed-with statement of the conformance statement of each PP-Module included in the components statement of the PP-Configuration shall identify all the PP-Configuration components that are not in the PP-Module Base(s) for that particular PP-Module as being allowed to be used in combination with the PP-Module in a PP-Configuration.*

11.9.1.3.18 Work unit ACE_CCO.1-17

For the exact conformance case, for each PP-Module listed in the PP-Configuration's components statement, the evaluator ***shall check*** the PP-Module's conformance statement to determine that all the components of the PP-Configuration are listed as allowed to be used with that PP-Module in a PP-Configuration, with the exception that PP-Configuration components in the PP-Module Base do not have to appear in the PP-Module's allowed-with list.

CC Part 3 ACE_CCO.1.16C: *For PP-Configurations that are not of exact conformance type (i.e. for PP-Configurations of strict or demonstrable conformance type), the conformance statement of a PP-*

Configuration may include an Evaluation Methods and Evaluation Activities reference statement that identifies the set of CEM-derived Evaluation Methods and Evaluation Activities that are applicable to the PP-Configuration under evaluation.

11.9.1.3.19 Work unit ACE_CCO.1-18

The evaluator ***shall check*** the conformance statement of the PP-Configuration under evaluation to confirm that:

a) If any derived Evaluation Methods and Evaluation Activities are required by other items included in the PP-Configuration (e.g. a base PP or PP-Module), or required by other items to which the PP-Configuration claims conformance (e.g. packages), then these are all identified in the PP-Configuration under evaluation, along with any derived Evaluation Methods and Evaluation Activities that the PP-Configuration itself chooses to require;

b) the list of derived Evaluation Methods and Evaluation Activities is sufficiently structured and detailed to unambiguously identify and locate every member of the list;

c) if there is any overlap in the scope of the identified Evaluation Methods and Evaluation Activities (i.e. where an overlay exists as described in CC Part 4) then the rationale for the resulting set of Evaluation Methods and Evaluation Activities is applicable to the TOE as described by the PP-Configuration under evaluation.

The intention of this work unit is to ensure that when evaluating a TOE that claims conformance with the PP-Configuration under evaluation then the correct Evaluation Methods and Evaluation Activities can be used. This means that identification in the PP-Configuration need not list individual Evaluation Activities where these are unambiguously included in a listed Evaluation Method. Similarly, where multiple Evaluation Methods or Evaluation Activities are included in a single document then it is sufficient to reference the document, provided this does lead to unambiguous identification of the Evaluation Methods and Evaluation Activities that apply to the PP-Configuration under evaluation.

EXAMPLE If a document lists multiple different Evaluation Methods applicable to different use cases then it would not be sufficient to reference the document: the relevant use cases would also have to be identified.

Where exact conformance is required then the PP-Configuration is not permitted to define its own requirements for derived Evaluation Methods and Evaluation Activities: it can only use those required by the other items (e.g. a base PP or PP-Module) in the PP-Configuration.

CC Part 3 ACE_CCO.1.17C: *The consistency rationale shall demonstrate that the TOE type defined in the PP-Configuration is consistent with the TOE types defined in the PPs and PP-Modules that belong to the PP-Configuration components statement.*

11.9.1.3.20 Work unit ACE_CCO.1-19

The evaluator ***shall examine*** the consistency rationale to determine that the TOE type defined in the PP-Configuration is consistent with the TOE types defined in the components of the PP-Configuration.

CC Part 3 ACE_CCO.1.18C: *The consistency rationale shall demonstrate that the union of all the SPDs, security objectives and security functional requirements defined in the PP-Configuration components is consistent.*

11.9.1.3.21 Work unit ACE_CCO.1-20

The evaluator ***shall examine*** the PP-Configuration consistency rationale to determine that it demonstrates that the union of all the SPDs of the PPs and PP-Modules identified in the PP-Configuration's components statement is consistent.

11.9.1.3.22 Work unit ACE_CCO.1-21

The evaluator ***shall examine*** the PP-Configuration consistency rationale to determine that it demonstrates that the union of all the security objectives of the PPs and PP-Modules identified in the PP-Configuration's components statement is consistent. If the PP-Configuration is a Direct Rationale PP-Configuration (as determined in ACE_CCO.1-3) then the TOE objectives are not required in the consistency analysis.

11.9.1.3.23 Work unit ACE_CCO.1-22

The evaluator ***shall examine*** the PP-Configuration consistency rationale to determine that the union of all the security functional requirements of the PPs and PP-Modules identified in the PP-Configuration's components statement is consistent, i.e. no contradiction arises from the whole set of SFRs of the PP-Configuration and its components.

CC Part 3 ACE_CCO.1.19C: *For a single-assurance PP-Configuration, the SAR statement shall define a single set of SARs that applies to the entire TOE. For strict and demonstrable conformance, the set of SARs shall include the SARs identified in each of the PP-Configuration components. For exact conformance, the set of SARs shall be identical to the set of SARs identified in each of the PP-Configuration components.*

11.9.1.3.24 Work unit ACE_CCO.1-23

The evaluator ***shall examine*** the PP-Configuration SAR statement to determine that it defines all applicable assurance requirements drawn from CC Part 3 and possibly extended. The set of SARs can refer to a well-defined SAR package, given in an external reference.

11.9.1.3.25 Work unit ACE_CCO.1-24

For a strict or demonstrable conformance PP-Configuration, the evaluator **shall check** that the set of SARs is well-formed: it is closed by dependencies or the SAR statement provides a sound discarding rationale.

11.9.1.3.26 Work unit ACE_CCO.1-25

For a strict or demonstrable conformance PP-Configuration, the evaluator **shall check** that the set of SARs of the PP-Configuration is consistent with respect to the SARs of each of the PPs and PP-Modules contained in the PP-Configuration: for any SAR component in each of the PPs and PP-Modules, the PP-Configuration provides either the same or a higher SAR component in the family hierarchy. If the SAR component in the PP/PP-Module is a refinement of a standard component, then the correspondent SAR component in the PP-Configuration has to include these refinements. If two PP-Configuration components refine the same SAR component, the evaluator ***shall check*** that the refinements are not contradictory and that the corresponding SAR component in the PP-Configuration meets both.

11.9.1.3.27 Work unit ACE_CCO.1-26

For an exact conformance PP-Configuration, the evaluator **shall check** that the set of SARs is identical to those SARs identified in each of the PP-Configuration components.

CC Part 3 ACE_CCO.1.20C: *For a multi-assurance PP-Configuration, the SAR statement shall define the global set of SARs that applies to the entire TOE and the SARs that apply to each sub-TSF. For strict and demonstrable conformance, the global assurance set of SARs shall include the set of common SARs among the PP-Configuration components, and each set of SARs that apply to a sub-TSF shall include those identified for the PP-Configuration components associated with that sub-TSF. For exact conformance, the global assurance set of SARs shall be the set of common SARs among the PP-Configuration components, and each set of SARs that apply to a sub-TSF shall be identical to those identified for the PP-Configuration components associated with that sub-TSF.*

11.9.1.3.28 Work unit ACE_CCO.1-27

For a multi-assurance PP-Configuration, the evaluator ***shall check*** that the SAR statement defines all the applicable assurance requirements. The evaluator ***shall examine*** the SAR statement to determine that it defines the global set of SARs applying to the entire TOE and the SARs applying to each sub-TSF.

11.9.1.3.29 Work unit ACE_CCO.1-28

For PP-Configurations of strict and/or demonstrable conformance, the evaluator shall examine the global set of SARs to ensure that it includes at least the set of SARs that are common to all the PP-Configuration components, and that any augmentation of this set is clearly identified.

In most cases (and always in the exact conformance case), the global set of SARs can be built as the common set of SARs that apply to all of the sub-TSFs. However, as it is the case with STs in the general model, the PP-Configuration (of strict or demonstrable conformance type) can require additional or higher SARs compared to the original SARs in the PP-Configuration components. The evaluation of the PP-Configuration will ensure the consistency of the claim, similar to the general model for the compliance with two or more PPs defining different sets of SARs, and similar to the approach for a multi-assurance ST which can extend the sets of SARs defined in the PP-Configuration the ST claims conformance to.

See CC Part 1, Clause 11.3.2.4, for further guidance and examples.

11.9.1.3.30 Work unit ACE_CCO.1-29

For PP-Configurations of exact conformance, the evaluator ***shall examine*** the global set of SARs to ensure that it includes only the set of SARs that are common to all the PP-Configuration components.

In the exact conformance case, the global set of SARs is built as the common set of SARs that apply to all of the sub-TSFs (with no augmentation or extension in the global set).

See CC Part 1, Clause 11.3.2.4, for further guidance and examples.

11.9.1.3.31 Work unit ACE_CCO.1-30

For PP-Configurations of strict and/or demonstrable conformance, the evaluator ***shall determine*** that each set of SARs that apply to a sub-TSF include those SARs associated with that sub-TSF's PP-Configuration components, but may include an augmentation of those SARs.

See CC Part 1, Clause 11.3.2.4, for further guidance and examples.

11.9.1.3.32 Work unit ACE_CCO.1-31

For PP-Configurations of exact conformance, the evaluator ***shall determine*** that each set of SARs that apply to a sub-TSF is identical those SARs associated with that sub-TSF's PP-Configuration components

CC Part 3 ACE_CCO.1.21C: *The SAR statement of a PP-Configuration shall include an assurance rationale that demonstrates the consistency of the applicable set of SARs with those defined in the components of the PP-Configuration under evaluation and their associated Evaluation Methods and Evaluation Activities. For a multi-assurance PP-Configuration, the assurance rationale shall demonstrate:*

- *that the global set of SARs is consistent with the threats as defined in the SPDs of the PP-Configuration components, and*
- *that the global set of SARs and the sets of SARs for each sub-TSF are consistent with each other.*

11.9.1.3.33 Work unit ACE_CCO.1-32

The evaluator ***shall check*** that the SAR statement of the PP-Configuration includes an assurance rationale.

11.9.1.3.34 Work unit ACE_CCO.1-33

The evaluator ***shall examine*** the assurance rationale to determine that the set of applicable SARs is consistent with those defined in the components of the PP-Configuration under evaluation.

11.9.1.3.35 Work unit ACE_CCO.1-34

The evaluator ***shall examine*** the assurance rationale to determine that the set of applicable SARs is consistent with the Evaluation Activities/Evaluation Methods associated with the PP-Configuration components.

11.9.1.3.36 Work unit ACE_CCO.1-35

For a multi-assurance PP-Configuration, the evaluator ***shall examine*** the assurance rationale to determine that the global set of SARs is consistent with the threats defined in the SPDs of the PP-Configuration components.

11.9.1.3.37 Work unit ACE_CCO.1-36

The evaluator ***shall check*** that the global set of SARs is consistent with all the sets of SARs for the sub-TSFs.

11.9.1.4 Action ACE_CCO.1.2E

11.9.1.4.1 Work unit ACE_CCO.1-37

The evaluator ***shall check*** that the PP-Configuration and the PP-Configuration components are consistent. That is, the evaluator shall check that no contradiction arises from the whole set of PP-Configuration components.

The evaluator can organise this work in many ways; the actual organisation may depend on the will to derive evaluation results for more than one PP-Configuration at a time.

For instance, the evaluator can proceed in two steps as follows:

a) assess the consistency of the PP-Configuration components;

b) then proceed with the assessment of the PP-Configuration consistency incrementally, by adding one PP-Module at a time, following the PPs/PP-Modules dependency structure.

An alternative is to proceed incrementally but mixing PPs and PP-Modules or to flatten/serialise the definition of the PP-Configuration (CC Part 1, Annex C), duplicating as required, and to assess the consistency of the whole set of elements.

Any incremental consistency analysis step where C is a subset of the PP-Configuration and X is a PP or a PP-Module that has to be added to C consists of:

- assessing that the SPD, the objectives and the SFRs of X do not contradict the statements in C;

- the assumptions and objectives for the environment in X either are the same as in C or address security aspects that are out of the scope of C.

If the PP-Configuration is a Direct Rationale PP-Configuration (as determined in ACE_CCO.1-3) then the TOE objectives are not required in the consistency analysis.

Note that if X is a PP-Module, C contains its PP-Module Base and Evaluation of sub-activity (ACE_MCO.1) has succeeded for X, then the consistency analysis step has to be performed with respect to the components of C that are not included in the PP-Module Base of X.

12 Class ASE: Security Target evaluation

12.1 General

This clause describes the evaluation of an ST. The ST evaluation should be started prior to any TOE evaluation sub-activities since the ST provides the basis and context to perform these sub-activities. The evaluation methodology in this subclause is based on the requirements on the ST as specified in CC Part 3 class ASE.

This clause should be used in conjunction with CC Part 1 Annexes A, B and C, as these Annexes clarify the concepts here and provide many examples.

12.2 Application notes

12.2.1 Re-using the evaluation results of certified PPs

While evaluating an ST that is based on one or more certified PPs, it may be possible to re-use the fact that these PPs were certified. The potential for re-use of the result of a certified PP is greater if the ST does not add threats, OSPs, assumptions, security objectives and/or security requirements to those of the PP. If the ST contains much more than the certified PP, re-use may not be useful at all.

The evaluator is allowed to re-use the PP evaluation results by doing certain analyses only partially or not at all if these analyses or parts thereof were already done as part of the PP evaluation. While doing this, the evaluator should assume that the analyses in the PP were performed correctly.

An example would be where the PP contains a set of security requirements, and these were determined to be internally consistent during the PP evaluation. If the ST uses the exact same requirements, the consistency analysis does not have to be repeated during the ST evaluation. If the ST adds one or more requirements, or performs operations on these requirements, the analysis will have to be repeated. However, it may be possible to save work in this consistency analysis by using the fact that the original requirements are internally consistent. If the original requirements are internally consistent, the evaluator only has to determine that:

a) the set of all new and/or changed requirements is internally consistent, and;

b) the set of all new and/or changed requirements is consistent with the original requirements.

The evaluator notes in the ETR each case where analyses are not done or only partially done for this reason.

The same re-use discussion applies to an ST claiming conformance to a certified PP-Configuration.

12.3 ST introduction (ASE_INT)

12.3.1 Evaluation of sub-activity (ASE_INT.1)

12.3.1.1 Objectives

The objective of this sub-activity is to determine whether the ST and the TOE are correctly identified, whether the TOE is correctly described in a narrative way at three levels of abstraction (TOE reference, TOE overview and TOE description), and whether these three descriptions are consistent with each other.

12.3.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.3.1.3 Action ASE_INT.1.1E

12.3.1.3.1 General

CC Part 3 ASE_INT.1.1C: *The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.*

12.3.1.3.2 Work unit ASE_INT.1-1

The evaluator ***shall check*** that the ST introduction contains an ST reference, a TOE reference, a TOE overview and a TOE description.

CC Part 3 ASE_INT.1.2C: *The ST reference shall uniquely identify the ST.*

12.3.1.3.3 Work unit ASE_INT.1-2

The evaluator ***shall examine*** the ST reference to determine that it uniquely identifies the ST.

The evaluator determines that the ST reference identifies the ST itself, so that it may be easily distinguished from other STs, and that it also uniquely identifies each version of the ST, e.g. by including a version number and/or a date of publication.

In evaluations where a CM system is provided, the evaluator may validate the uniqueness of the reference by checking the configuration list. In the other cases, the ST should have some referencing system that is capable of supporting unique references (e.g. use of numbers, letters or dates).

CC Part 3 ASE_INT.1.3C: *The TOE reference shall uniquely identify the TOE.*

12.3.1.3.4 Work unit ASE_INT.1-3

The evaluator ***shall examine*** the TOE reference to determine that it uniquely identifies the TOE.

The evaluator determines that the TOE reference uniquely identifies the TOE, so that it is clear to which TOE the ST refers, and that it also identifies the version of the TOE, e.g. by including a version/release/build number, or a date of release.

In the end of the evaluation, the evaluator ***shall check*** the TOE reference, and any unique identifiers associated with the TOE physical components are consistent with the identifier(s) assigned to the TOE evaluated in work units related to ALC_CMC.x.1C and the configuration list evaluated in work units related to ALC_CMS.x.2C.

12.3.1.3.5 Work unit ASE_INT.1-4

The evaluator ***shall examine*** the TOE reference to determine that it is not misleading.

If the TOE is related to one or more well-known products, it is allowed to reflect this in the TOE reference. However, this should not be used to mislead consumers and it must be made clear which part of the product has been evaluated.

When a TOE needs some required non-TOE hardware/software/firmware to run properly, the TOE reference may include the name of the non-TOE hardware/software/firmware used by the TOE, however it must be made clear that the non-TOE hardware/software/firmware has not been evaluated.

CC Part 3 ASE_INT.1.4C: *The TOE overview shall summarize the usage and major security features of the TOE.*

12.3.1.3.6 Work unit ASE_INT.1-5

The evaluator ***shall examine*** the TOE overview to determine that it describes the usage and major security features of the TOE.

The TOE overview may describe security features that are provided by the product, and/or those that users may expect in that product type, but it must clearly distinguish those features that are evaluated and those that are not evaluated.

The TOE overview shall be consistent with information provided in other sections of the Security Target such as the TOE description, the security objectives, the security functional requirements, and the TOE summary specification. In addition to ensuring the evaluated security features are consistently described throughout the ST, this means that any security feature that is not evaluated is only discussed within the ST introduction, or else is explicitly identified as not evaluated in each other place where it is mentioned (failure to make this identification means that this work unit is assigned a fail verdict).

The TOE overview in an ST for a composed TOE should describe the usage and major security feature of the composed TOE, rather than those of the individual component TOEs.

The evaluator determines that the overview is clear enough for consumers, and sufficient to give them a general understanding of the intended usage and major security features of the TOE.

CC Part 3 ASE_INT.1.5C: *The TOE overview shall identify the TOE type.*

12.3.1.3.7 Work unit ASE_INT.1-6

The evaluator ***shall check*** that the TOE overview identifies the TOE type.

12.3.1.3.8 Work unit ASE_INT.1-7

The evaluator ***shall examine*** the TOE overview to determine that the TOE type is not misleading.

There are situations where the general consumer would expect certain functionality of the TOE because of its TOE type. If this functionality is absent in the TOE, the evaluator determines that the TOE overview adequately discusses this absence.

There are also TOEs where the general consumer would expect that the TOE should be able to operate in a certain operational environment because of its TOE type. If the TOE is unable to operate in such an operational environment, the evaluator determines that the TOE overview adequately discusses this.

CC Part 3 ASE_INT.1.6C: *The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.*

12.3.1.3.9 Work unit ASE_INT.1-8

The evaluator ***shall examine*** the TOE overview to determine that it identifies any non-TOE hardware/software/firmware required by the TOE.

While some TOEs are able to run stand-alone, other TOEs (notably software TOEs) need additional hardware, software or firmware to operate. If the TOE does not require any hardware, software or firmware, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the TOE overview identifies any additional hardware, software and firmware needed by the TOE to operate. This identification does not have to be exhaustive,

but detailed enough for potential consumers of the TOE to determine whether their current hardware, software and firmware support use of the TOE, and, if this is not the case, which additional hardware, software and/or firmware is needed.

CC Part 3 ASE_INT.1.7C: *For a multi-assurance ST, the TOE overview shall describe the TSF organization in terms of the sub-TSFs defined in the PP-Configuration the ST claims conformance to.*

12.3.1.3.10 Work unit ASE_INT.1-9

For a multi-assurance ST, the evaluator ***shall examine*** the TOE overview to determine that it describes the TSF organization in terms of the sub-TSFs defined in the PP-Configuration to which the ST claims conformance. The TSF organization is possibly completed with details of the actual TOE.

CC Part 3 ASE_INT.1.8C: *The TOE description shall describe the physical scope of the TOE.*

12.3.1.3.11 Work unit ASE_INT.1-10

The evaluator ***shall examine*** the TOE description to determine that it describes the physical scope of the TOE.

The evaluator determines that the TOE description lists the hardware, firmware, software and guidance parts that constitute the TOE and describes them at a level of detail that is sufficient to give the reader a general understanding of those parts.

As a minimum, the TOE description will cover the following elements:

a) each separately delivered part of the TOE, which will be identified by its unique identifier and the current format (binary, wafer, inlay, *.pdf, *.doc, *.chm etc.);

b) the delivery method used by the developer to make available each part to the TOE consumer (web site download, courier delivery, etc.).

The physical description will also include some clear statements about the evaluated TOE configuration. In the case where a product can have multiple physical components, and therefore multiple configurations, the evaluated configurations must be briefly described and identified.

The evaluator also determines that there is no possible misunderstanding as to whether any hardware, firmware, software or guidance part is part of the TOE or not.

CC Part 3 ASE_INT.1.9C: *The TOE description shall describe the logical scope of the TOE.*

12.3.1.3.12 Work unit ASE_INT.1-11

The evaluator ***shall examine*** the TOE description to determine that it describes the logical scope of the TOE.

The evaluator determines that the TOE description discusses the logical security features offered by the TOE at a level of detail that is sufficient to give the reader a general understanding of those features.

The evaluator also determines that there is no possible misunderstanding as to whether any logical security feature is offered by the TOE or not.

An ST for a composed TOE may refer out to the description of the logical scope of the component TOEs, provided in the component TOE STs to provide the majority of this description for the composed TOE. However, the evaluator determines that the composed TOE ST clearly discusses

which features of the individual components are not within the composed TOE, and therefore not a feature of the composed TOE.

12.3.1.4 Action ASE_INT.1.2E

12.3.1.4.1 Work unit ASE_INT.1-12

The evaluator ***shall examine*** the TOE reference, TOE overview and TOE description to determine that they are consistent with each other.

12.4 Conformance claims (ASE_CCL)

12.4.1 Evaluation of sub-activity (ASE_CCL.1)

12.4.1.1 Objectives

The objective of this sub-activity is to determine the validity of various conformance claims. These describe how the ST and the TOE conform to the CC and how the ST conforms to a PP-Configuration, PPs and packages.

12.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the PP(s) or PP-Configuration that the ST claims conformance to;

c) the package(s) that the ST claims conformance to.

12.4.1.3 Action ASE_CCL.1.1E

12.4.1.3.1 General

CC Part 3 ASE_CCL.1.1C: *The conformance claim shall identify the edition of the CC to which the ST and the TOE claim conformance.*

12.4.1.3.2 Work unit ASE_CCL.1-1

The evaluator ***shall check*** that the conformance claim identifies the edition of the CC to which the ST and the TOE claim conformance.

The evaluator determines that the conformance claim identifies the edition of the CC that was used to develop this ST. This should include the version number of the CC and, unless the English version of the CC was used, the language of the edition of the CC that was used.

For a composed TOE, the evaluator will consider any differences between the edition of the CC claimed for a component and the edition of the CC claimed for the composed TOE. If the editions differ the evaluator will assess whether the differences between them will lead to conflicting claims.

For instances where conformance claims for the base TOE and dependent TOE are for different major releases of the CC (e.g. one component TOE conformance claim is CC v2.x and the other component TOE conformance claim is CC v3.x), the conformance claim for the composed TOE will be the earlier release of the CC, as the CC is developed with an aim to provide backwards compatibility (although this may not be achieved in the strictest sense, it is understood to be achieved in principle).

CC Part 3 ASE_CCL.1.2C: *The conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.*

12.4.1.3.3 Work unit ASE_CCL.1-2

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 2 conformant or CC Part 2 extended for the ST.

For a composed TOE, the evaluator will consider whether this claim is consistent not only with CC Part 2, but also with the claims of conformance to CC Part 2 by each of the component TOEs, i.e. if one or more component TOEs claims to be CC Part 2 extended, then the composed TOE should also claim to be CC Part 2 extended.

The conformance claim for the composed TOE may be CC Part 2 extended, even though the component TOEs are CC Part 2 conformant, in the event that additional SFRs are claimed for the base TOE (see composed TOE guidance for ASE_CCL.1.6C)

CC Part 3 ASE_CCL.1.3C: *The conformance claim shall describe the conformance of the ST as either "CC Part 3 conformant" or "CC Part 3 extended".*

12.4.1.3.4 Work unit ASE_CCL.1-3

The evaluator ***shall check*** that the conformance claim states a claim of either CC Part 3 conformant or CC Part 3 extended for the ST.

CC Part 3 ASE_CCL.1.4C: *The conformance claim shall be consistent with the extended components definition.*

12.4.1.3.5 Work unit ASE_CCL.1-4

The evaluator ***shall examine*** the conformance claim for CC Part 2 to determine that it is consistent with the extended components definition.

If the conformance claim contains CC Part 2 conformant, the evaluator determines that the extended components definition does not define functional components.

If the conformance claim contains CC Part 2 extended, the evaluator determines that the extended components definition defines at least one extended functional component.

12.4.1.3.6 Work unit ASE_CCL.1-5

The evaluator ***shall examine*** the conformance claim for CC Part 3 to determine that it is consistent with the extended components definition.

If the conformance claim contains CC Part 3 conformant, the evaluator determines that the extended components definition does not define assurance components.

If the conformance claim contains CC Part 3 extended, the evaluator determines that the extended components definition defines at least one extended assurance component.

CC Part 3 ASE_CCL.1.5C: *The conformance claim shall identify a PP-Configuration, or all PPs and security requirement packages to which the ST claims conformance.*

12.4.1.3.7 Work unit ASE_CCL.1-6

The evaluator ***shall check*** that the conformance claim contains a PP claim that identifies all PPs for which the ST claims conformance.

If the ST does not claim conformance to a PP, this work unit is not applicable and therefore considered to be satisfied.

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).

12.4.1.3.8 Work unit ASE_CCL.1-7

For conformance claims to PPs containing functional packages, the evaluator ***shall check*** that:

- for each PP to which the ST claims conformance, the ST does not include conformance claims to any packages to which the PP also claims conformance, unless the ST augments the requirements in that package. For example, if a PP claims <package foo>-conformant to package foo, and an ST claims conformance to the PP, then the ST does not also claim "<package foo>-conformant" in its conformance claim. However, if it augments package foo with a requirement, then it would claim "<package foo>-augmented" in its conformance claim.

The evaluator determines that if the ST does claim conformance to a package that is claimed by one of the PPs to which the ST is claiming conformance, the PP has a conformance type of strict or demonstrable. An ST is not allowed to claim conformance on any functional packages when claiming exact conformance to one or more PPs.

The evaluator is reminded that claims of partial conformance to a PP are not permitted. Therefore, conformance to a PP requiring a composite solution may be claimed in an ST for a composed TOE. Conformance to such a PP would not have been possible during the evaluation of the component TOEs, as these components would not have satisfied the composed solution. This is only possible in the instances where the "composite" PP permits use of the composition evaluation approach (use of ACO components).

For PPs containing functional packages, partial conformance means that not all packages have been included in the ST, a functional package has only been partially included into the ST, or a dependency requirement within or between functional packages has not been met. Note that exclusion of optional requirements that the ST either chooses not to, or is not required to, claim does not result in "partial conformance" to the PP, and is therefore allowed.

12.4.1.3.9 Work unit ASE_CCL.1-8

The evaluator ***shall check*** that, for each PP to which the ST claims conformance, the allowed-with statement included in the conformance statement of that PP lists all other PPs in the ST's conformance claim.

If the ST is not claiming exact conformance to more than one PP, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the allowed-with statement included in the conformance statement of each PP to which conformance is being claimed lists each of the other PPs identified in the conformance claim section of the ST as being "allowed to be claimed with" that PP. Note that this is only applicable in cases where that PP requires exact conformance and the ST claims exact conformance.

EXAMPLE Consider the case where an ST is being evaluated and claims conformance to PPs B and C; this is depicted in Figure 8. The ST is claiming exact conformance, so all PPs require exact conformance in their conformance statements. Under this work unit, the evaluator determines that PP B lists (in its allowed-with statement) "PP C" as being a PP that can be claimed (by an ST) with PP B. Likewise, the evaluator determines that PP C lists (in its allowed-with statement) "PP B" as being a PP that can be claimed (by an ST) with PP C.

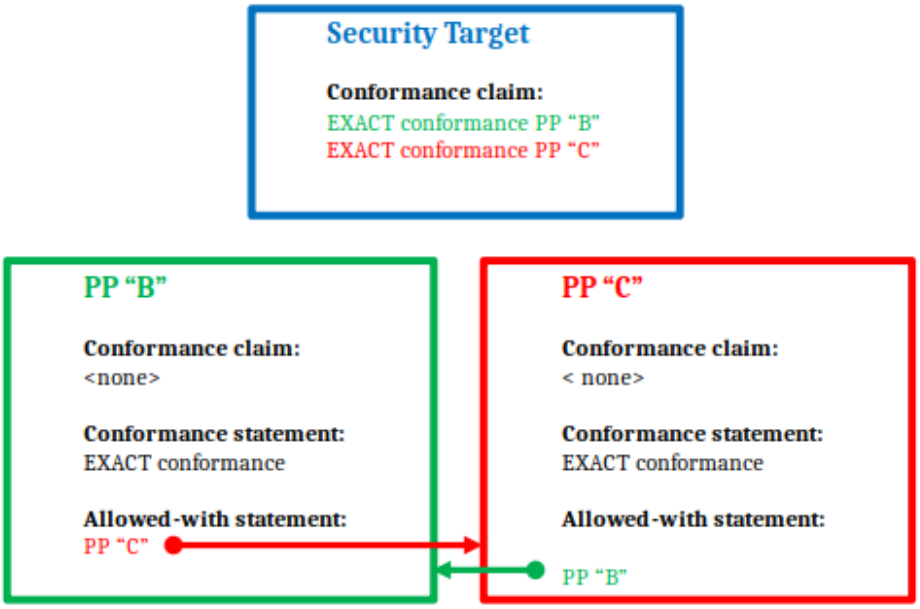

Figure 8 — Example of exact conformance relationships between an ST and PPs

12.4.1.3.10 Work unit ASE_CCL.1-9

The evaluator ***shall check*** that the conformance claim contains a PP-Configuration claim that identifies the PP-Configuration to which the ST claims conformance.

If the ST does not claim conformance to a PP-Configuration, this work unit is not applicable and therefore considered to be satisfied.

12.4.1.3.11 Work unit ASE_CCL.1-10

The evaluator ***shall check*** that the ST claims conformance to exactly one PP-Configuration.

An ST cannot simultaneously claim conformance to a PP-Configuration and one or more PP(s) that is/are not part of that PP-Configuration. The evaluator determines that the referenced PP-Configuration is unambiguously identified (e.g. by title and version number, or by the identification included in the introduction of that PP-Configuration).

An ST is not allowed to claim conformance to a PP-Configuration and a functional package, so the evaluator should confirm that no functional package conformance claims are included in the ST conformance claim section if it claims conformance to a PP-Configuration.

If an ST claims conformance to a PP-Configuration that claims, or whose components claim, conformance to a security assurance package, the evaluator ensures that the ST does not include conformance claims to those packages unless the ST augments those packages.

12.4.1.3.12 Work unit ASE_CCL.1-11

The evaluator ***shall check***, for each identified functional package, that the package definition is complete.

If the ST does not claim conformance to a functional package, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 "Packages" by checking that the functional package includes:

a) A functional package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) A functional package overview, giving a narrative description of the security functionality;

c) A component rationale that provides the rationale for selecting the functional components/requirements included in the package;

d) If the package defines an SPD then:

   i. the package includes a security requirements rationale;

   ii. the package includes a security objectives rationale if security objectives for the environment are defined;

   iii. If the package is a direct rationale package, there are no security objectives for the TOE defined and the security requirements rationale maps directly to the SPD;

   iv. If the package is not a direct rationale package, then security objectives for the TOE are defined, the security objectives rationale covers the objectives with respect to the SPD, and the security requirements rationale maps the requirements to the security objectives.

e) one or more security components or requirements (the functional package SFRs);

f) If extended components have been specified then the functional package includes an extended components definition;

12.4.1.3.13 Work unit ASE_CCL.1-12

The evaluator ***shall check***, for each identified assurance package, that the package definition is complete. If the ST does not claim conformance to an assurance package, this work unit is not applicable and therefore considered to be satisfied. If the assurance package is a reference to one of the assurance packages contained in CC Part 5 then this work unit is also considered to be satisfied. The evaluator determines that the package definition is conformant to the requirements from CC Part 1, Clause 9 “Packages” by checking that the assurance package includes:

a) An assurance package identification, giving a unique name, version, date, sponsor, and the CC edition;

b) An assurance package overview, giving a narrative description of the security functionality;

c) One or more security components or requirements (the assurance package SARs) drawn from CC Part 3, extended assurance components or some combination of both;

d) An assurance package shall not include an SPD or security objectives;

e) If extended components have been specified then the assurance package includes an extended components definition;

A security requirements rationale that provides the rationale for selecting the assurance components/ requirements included in the package.

CC Part 3 ASE_CCL.1.6C: *The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.*

12.4.1.3.14 Work unit ASE_CCL.1-13

The evaluator ***shall check*** that the conformance claim contains a package claim that identifies all packages to which the ST claims conformance.

If the ST does not claim conformance to a package, this work unit is not applicable and therefore considered to be satisfied.

Stipulations on packages to which the ST claims conformance that are also claimed conformance to by a PP to which the ST is claiming conformance to are addressed in other work units. The evaluator also determines that if the ST is claiming conformance to a PP-Configuration, then only assurance packages are claimed conformance to by the ST. Additionally, the evaluator determines that if the ST is claiming exact conformance to a PP or PP-Configuration, then no packages are claimed conformance to by the ST.

The evaluator is reminded that claims of partial conformance to a package are not permitted.

12.4.1.3.15 Work unit ASE_CCL.1-14

The evaluator ***shall check*** that, for each identified package, the conformance claim states a claim of either package-conformant or package-augmented.

If the ST claims conformance to a PP/PP-Configuration and the PP/PP-Configuration component itself claims conformance to one or more functional packages then the ST shall not separately make a conformance claim to the same packages unless the conditions outlined in work unit ASE_CCL.1-8 (for PPs) or ASE_CCL.1-9 (for PP-Configuration components) are met. If the ST does not claim conformance to a package, this work unit is not applicable and therefore considered to be satisfied.

If the package conformance claim contains package-conformant, the evaluator determines that:

a) if the package is an assurance package, then the ST contains all SARs included in the package, but no additional SARs;

b) if the package is a functional package, then all assumptions, threats, OSPs, security objectives and SFRs included in the package are identical to those included in the ST (after allowing any remaining iterations, refinements, assignments or selections from the package to be made in the ST).

If the package conformance claim contains package-augmented, the evaluator determines that:

a) if the package is an assurance package then the ST contains all SARs included in the package, and at least one additional SAR or at least one SAR that is hierarchical to a SAR in the package;

b) if the package is a functional package, then the constituent parts (security problem definition, security objectives, SFRs) of that ST contain all constituent parts (security problem definition, security objectives, SFRs) of that specific package, but additionally contain at least one enhancement of the security functionality defined by that specific package (finally resulting in an additional SFR or one an SFR that is hierarchically higher than an SFR in the package).

The evaluator determines that, if the ST claims exact conformance to a PP/PP-Configuration, no package conformance claims are present.

CC Part 3 ASE_CCL.1.7C: *The conformance claim shall describe any conformance of the ST to a PP as PP-Conformant.*

12.4.1.3.16 Work unit ASE_CCL.1-15

The evaluator ***shall check*** that if the ST claims conformance to PPs, only PP-Conformant claims are present. For a Direct Rationale ST, the evaluator ***shall check*** that only conformance claims to Direct Rationale PPs are present.

CC Part 3 ASE_CCL.1.8C: *The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PP-Configuration or PPs for which conformance is being claimed.*

12.4.1.3.17 Work unit ASE_CCL.1-16

In this work unit, the term "PP" shall be understood to mean "PP or PP-Configuration component".

The evaluator ***shall examine*** the conformance claim rationale to determine that the TOE type of the TOE is consistent with all TOE types of the PPs.

If the ST does not claim conformance to a PP, this work unit is not applicable and therefore considered to be satisfied.

The relation between the types may be simple: a firewall ST claiming conformance to a firewall PP, or more complex: a smart card ST claiming conformance to a number of PPs at the same time (a PP for the integrated circuit, a PP for the smart card OS, and two PPs for two applications on the smart card).

For a composed TOE, the evaluator will determine whether the conformance claim rationale demonstrates that the TOE types of the component TOEs are consistent with the composed TOE type. This does not mean that both the component and the composed TOE types have to be the same, but rather that the component TOEs are suitable for integration to provide the composed TOE. It should be made clear in the composed TOE ST which SFRs are only included as a result of composition, and were not examined as SFRs in the base and dependent TOE (e.g. EALx) evaluation.

CC Part 3 ASE_CCL.1.9C: *The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PP-Configuration*[1]*, PPs and any functional packages for which conformance is being claimed.*

12.4.1.3.18 Work unit ASE_CCL.1-17

The evaluator ***shall examine*** the conformance claim rationale to determine that it demonstrates that the statement of security problem definition is consistent with the statements of security problem definition stated in the PPs/PP-Configuration/functional packages to which conformance is being claimed.

If the ST does not claim conformance with any PP/PP-Configuration/functional package, this work unit is not applicable and therefore considered to be satisfied.

If the PP/PP-Configuration/functional package does not have a statement of security problem definition, this work unit is not applicable and therefore considered to be satisfied.

If the PP/PP-Configuration contains functional packages, the evaluator determines that the security problem definition of the ST consists of all assumptions, threats and OSPs of all functional packages.

---

1 In practice, this refers to the union of SPDs defined in the PP-Configuration components.

If functional packages are claimed, the rules defined in the following paragraphs concerning exact, strict and demonstrable conformance also hold for the SPD descriptions taken from the functional packages.

If exact conformance is required by the PPs/PP-Configuration to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether:

a) the threats in the ST are identical (no fewer threats, no additional threats) to the threats in the PP/PP-Configuration to which conformance is being claimed. If exact conformance is being claimed to more than one PP, then the set of threats in the ST must be identical to the union of the threats in all PPs to which conformance is being claimed.

b) the OSPs in the ST are identical (no fewer OSPs, no additional OSPs) to the OSPs in the PP/PP-Configuration to which conformance is being claimed. If exact conformance is being claimed to more than one PP, then the set of OSPs in the ST must be identical to the union of the OSPs in all PPs to which conformance is being claimed.

c) the assumptions in the ST are identical (no fewer assumptions, no additional assumptions) to the assumptions in the PP/PP-Configuration to which conformance is being claimed. If exact conformance is being claimed to more than one PP, then the set of assumptions in the ST must be identical to the union of the assumptions in all PPs to which conformance is being claimed, with the following possible exception;

An assumption (or part of an assumption) from a PP can be omitted, if all security objectives for the operational environment addressing this assumption (or part of an assumption) are replaced by security objectives for the TOE that are identical to (taken from) another of the PPs to which the ST is claiming conformance.

When examining an ST in these circumstances (assumptions from one PP are replaced by security objectives on the TOE from one of the other PPs) the evaluator shall carefully determine that the condition given above is fulfilled. The following discussion gives an example:

EXAMPLE An ST is claiming exact conformance to two PPs. As determined in previous work units, both PPs require exact conformance in their conformance statements, and both PPs list the other as being "allowed with" the PP in a conformance claim by an ST. One PP to which the ST claims conformance contains an assumption stating that the operational environment prevents unauthorized modification or interception of data sent to an external interface of the TOE. This may be the case if the TOE accepts data in clear text and without integrity protection at this interface and is assumed to be located in a secure operational environment, which will prevent attackers from accessing this data. The assumption will then be mapped in the PP to some objective for the operational environment stating that the data interchanged at this interface are protected by adequate measures in the operational environment. Suppose there is another PP that specifies that conformant TOEs must protect data sent over the TOEs external interfaces, and has appropriate threats and security objectives addressing this threat. The ST author can then replace the assumption and security objective for the environment related to the protection of data over the external interfaces of the TOE from one PP with the security objective stating that the TOE itself protects these data, for example by providing a secure channel for encryption and integrity protection of all data transferred via this interface from the other PP; the corresponding objective and assumption for the operational environment from the other PP is thus omitted from the ST. This is also called re-assigning of the objective, since the objective is re-assigned from the operational environment to the TOE. Note, that this TOE is still secure in an operational environment fulfilling the omitted assumption and therefore still fulfils the PP. Further, the set of threats and objectives in the ST is still no broader than the union of threats and objectives in the PPs to which it is claiming exact conformance.

If strict conformance is required by the PPs/PP-Configuration to which conformance is being claimed no conformance claim rationale is required. Instead, the evaluator determines whether:

a) the threats in the ST are a superset of or identical to the threats in the PPs/PP-Configuration to which conformance is being claimed;

b) the OSPs in the ST are a superset of or identical to the OSPs in the PPs/PP-Configuration to which conformance is being claimed;

c) the assumptions in the ST are identical to the assumptions in the PPs/PP-Configuration to which conformance is being claimed, with two possible exceptions described in the following two bullet points:

- an assumption (or part of an assumption) from a PP/PP-Configuration can be omitted, if all security objectives for the operational environment addressing this assumption (or part of an assumption) are replaced by security objectives for the TOE;

- an assumption can be added to the assumptions defined in a PP/PP-Configuration, if a rationale is given, why the new assumption neither mitigates a threat (or a part of a threat) meant to be addressed by security objectives for the TOE in the PP/PP-Configuration, nor fulfils an OSP (or part of an OSP) meant to be addressed by security objectives for the TOE in the PP/PP-Configuration.

When examining an ST claiming a PP/PP-Configuration, which omits assumptions from the PP/PP-Configuration or adds new assumptions, the evaluator shall carefully determine, if the conditions given above are fulfilled. The following discussion gives some motivation and examples for these cases:

- example for omitting an assumption: A PP may contain an assumption stating that the operational environment prevents unauthorized modification or interception of data sent to an external interface of the TOE. This may be the case if the TOE accepts data in clear text and without integrity protection at this interface and is assumed to be located in a secure operational environment, which will prevent attackers from accessing these data. The assumption will then be mapped in the PP to some objective for the operational environment stating that the data interchanged at this interface are protected by adequate measures in the operational environment. If an ST claiming this PP defines a more secure TOE, which has an additional security objective stating that the TOE itself protects these data, for example by providing a secure channel for encryption and integrity protection of all data transferred via this interface, the corresponding objective and assumption for the operational environment can be omitted from the ST. This is also called re-assigning of the objective, since the objective is re-assigned from the operational environment to the TOE. Note, that this TOE is still secure in an operational environment fulfilling the omitted assumption and therefore still fulfils the PP;

- example for adding an assumption: In this example, the PP is designed to specify requirements for a TOE of type "Firewall" and an ST author wishes to claim this PP for a TOE, which implements a firewall, but additionally provides the functionality of a virtual private network (VPN) component. For the VPN functionality, the TOE needs cryptographic keys and these keys may also have to be handled securely by the operational environment (e. g. if symmetric keys are used to secure the network connection and therefore need to be provided in some secure way to other components in the network). In this case, it is acceptable to add an assumption that the cryptographic keys used by the VPN are handled securely by the operational environment. This assumption does not address threats or OSPs of the PP and therefore fulfils the conditions stated above;

- counter example for adding an assumption: In a variant of the first example a PP may already contain an objective for the TOE to provide a secure channel for one of its interfaces, and

this objective is mapped to a threat of unauthorized modification or reading of the data on this interface. In this case, it is clearly not allowed for an ST claiming this PP to add an assumption for the operational environment, which assumes that the operational environment protects data on this interface against modification or unauthorized reading of the data. This assumption would reduce a threat, which is meant to be addressed by the TOE. Therefore, a TOE fulfilling an ST with this added assumption would not automatically fulfil the PP anymore and this addition is therefore not allowed;

- second counter example for adding an assumption: In the example above of a TOE implementing a firewall it would not be admissible to add a general assumption that the TOE is only connected to trusted devices, because this would obviously remove essential threats relevant for a firewall (namely that there is untrusted IP traffic, which needs to be filtered). Therefore, this addition would not be allowed.

If demonstrable conformance is required by the PP/PP-Configuration, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security problem definition of the ST is equivalent or more restrictive than the statement of security problem definition in the PP/PP-Configuration to which conformance is being claimed.

For this, the conformance claim rationale needs to demonstrate that the security problem definition in the ST is equivalent (or more restrictive) than the security problem definition in the PP/PP-Configuration. This means that:

- all TOEs that would meet the security problem definition in the ST also meet the security problem definition in the PP/PP-Configuration. This can also be shown indirectly by demonstrating that every event, which realises a threat defined in the PP or violates an OSP defined in the PP/PP-Configuration, would also realise a threat stated in the ST or violate an OSP defined in the ST. Note that fulfilling an OSP stated in the ST may avert a threat stated in the PP/PP-Configuration or that averting a threat stated in the ST may fulfil an OSP stated in the PP/PP-Configuration, so threats and OSPs can substitute each other;

- all operational environments that would meet the security problem definition in the PP/PP-Configuration would also meet the security problem definition in the ST (with one exception in the next bullet);

- besides a set of assumptions in the ST needed to demonstrate conformance to the SPD of the PP/PP-Configuration, an ST may specify further assumptions, but only if these additional assumptions are independent of and do not affect the security problem definition as defined in the PP/PP-Configuration. More detailed, there are no assumptions in the ST that exclude threats to the TOE that need to be countered by the TOE according to the PP. Similarly, there are no assumptions in the ST that realise aspects of an OSP stated in the PP/PP-Configuration, which are meant to be fulfilled by the TOE according to the PP/PP-Configuration.

For a composed TOE, the evaluator will consider whether the security problem definition of the composed TOE is consistent with that specified in the STs for the component TOEs. This is determined in terms of demonstrable conformance. In particular, the evaluator examines the conformance claim rationale to determine that:

a) threat statements and OSPs in the composed TOE ST do not contradict those from the component STs;

b) any assumptions made in the component STs are upheld in the composed TOE ST. That is, either the assumption should also be present in the composed ST, or the assumption should be positively addressed in the composed ST. The assumption may be positively addressed

through specification of requirements in the composed TOE to provide functionality fulfilling the concern captured in the assumption.

CC Part 3 ASE_CCL.1.10C: *The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PP-Configuration[2], PPs, and any functional package for which conformance is being claimed.*

12.4.1.3.19 Work unit ASE_CCL.1-18

The evaluator ***shall examine*** the conformance claim rationale to determine that the statement of security objectives is consistent, with the statement of security objectives in the PPs/PP-Configuration/functional packages to which conformance is being claimed.

If the ST does not claim conformance to a PP/PP-Configuration/functional packages, this work unit is not applicable and therefore considered to be satisfied.

If the PP/PP-Configuration to which conformance is being claimed contains functional packages, the evaluator determines that the security objectives of the ST consist of all security objectives of all functional packages.

If functional packages are claimed the rules defined in the following paragraphs concerning exact, strict and demonstrable conformance also hold for the security objectives taken from these packages.

If exact conformance is required by the PPs/PP-Configuration to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether:

a) the ST contains all security objectives for the TOE of the PP/PP-Configuration to which conformance is being claimed. Note that in the exact conformance case, it is not allowed for the ST under evaluation to have additional security objectives for the TOE. If conformance is being claimed to more than one PP, the set of security objectives for the TOE must be identical to the union of the security objectives for the TOE in the PPs to which conformance is being claimed. It should be noted that in the case that optional requirements have associated SPD elements, exact conformance can still be claimed if objectives associated with the SPD elements are omitted when the associated optional SFRs are also omitted;

b) the security objectives for the operational environment in the ST are identical to the security objectives for the operational environment in the PP/PP-Configuration to which conformance is being claimed. If conformance is being claimed to more than one PP, the set of security objectives for the operational environment must be identical to the union of the security objectives for the operational environment in the PPs to which conformance is being claimed with the possible exception as follows:

- a security objective for the operational environment (or part of such security objective) from one PP can be replaced by the same (part of the) security objective for the TOE from another PP.

If strict conformance is required by the PPs/PP-Configuration to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether:

[2] In practice, this refers to the union of security objectives defined in the PP-Configuration components.

- the ST contains all security objectives for the TOE of the PP/PP-Configuration to which conformance is being claimed. Note that it is allowed for the ST under evaluation to have additional security objectives for the TOE;

- the security objectives for the operational environment in the ST are identical to the security objectives for the operational environment in the PP/PP-Configuration to which conformance is being claimed, with two possible exceptions described in the following two bullet points;

- a security objective for the operational environment (or part of such security objective) from the PP/PP-Configuration can be replaced by the same (part of the) security objective stated for the TOE;

- a security objective for the operational environment can be added to the objectives defined in the PP/PP-Configuration, if a justification is given, why the new objective neither mitigates a threat (or a part of a threat) meant to be addressed by security objectives for the TOE in the PP/PP-Configuration, nor fulfils an OSP (or part of an OSP) meant to be addressed by security objectives for the TOE in the PP/PP-Configuration.

When examining an ST claiming a PP/PP-Configuration, which omits security objectives for the operational environment from the PP/PP-Configuration or adds new security objectives for the operational environment, the evaluator shall carefully determine, if the conditions given above are fulfilled. The examples given for the case of assumptions in the preceding work unit are also valid here.

If demonstrable conformance is required by the PPs/PP-Configuration to which conformance is being claimed, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security objectives of the ST is equivalent or more restrictive than the statement of security objectives in the PP/PP-Configuration to which conformance is being claimed.

For this the conformance claim rationale needs to demonstrate that the security objectives in the ST are equivalent (or more restrictive) than the security objectives in the PP/PP-Configuration. This means that:

- all TOEs that would meet the security objectives for the TOE in the ST also meet the security objectives for the TOE in the PP/PP-Configuration;

- all operational environments that would meet the security objectives for the operational environment in the PP/PP-Configuration would also meet the security objectives for the operational environment in the ST (with one exception in the next bullet);

- besides a set of security objectives for the operational environment in the ST, which are used to demonstrate conformance to the set of security objectives defined in the PP/PP-Configuration, an ST may specify further security objectives for the operational environment, but only if these security objectives neither affect the original set of security objectives for the TOE nor the security objectives for the operational environment as defined in the PP/PP-Configuration to which conformance is claimed.

For a composed TOE, the evaluator will consider whether the security objectives of the composed TOE are consistent with that specified in the STs for the component TOEs. This is determined in terms of demonstrable conformance. In particular, the evaluator examines the conformance claim rationale to determine that:

- the statement of security objectives in the dependent TOE ST relevant to any IT in the operational environment are consistent with the statement of security objectives for the TOE in the base TOE ST. It is not expected that the statement of security objectives for the environment within in the dependent TOE ST will cover all aspects of the statement of security objectives for the TOE in the base TOE ST;

- the statement of security objectives in the composed ST is consistent with the statements of security objectives in the STs for the component TOEs.

CC Part 3 ASE_CCL.1.11C: *The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PP-Configuration[3], PPs, and any functional packages for which conformance is being claimed.*

12.4.1.3.20 Work unit ASE_CCL.1-19

The evaluator ***shall examine*** the ST to determine that it is consistent with all security requirements in the PPs/PP-Configuration/functional packages for which conformance is being claimed.

If the ST does not claim conformance to a PP/PP-Configuration/functional package, this work unit is not applicable and therefore considered to be satisfied.

If the PPs/PP-Configuration to which conformance is being claimed contains functional packages, the evaluator determines that the SFRs of the ST consist of all SFRs (or hierarchical SFRs) of all functional packages.

If functional packages are used, the rules defined in the following paragraphs concerning exact, strict and demonstrable conformance also hold for the SFRs taken from these packages.

If exact conformance is required by the PP/PP-Configuration to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines that the statement of security requirements in the PP to which conformance is being claimed is exactly reproduced in the ST, with the following allowances:

a) an SFR from the PP/PP-Configuration may be iterated or refined in the ST;

b) all SFRs that are defined in the PP/PP-Configuration to which conformance is being claimed as selection-based upon a particular selection shall be included if and only if that selection on which inclusion is based is present in the ST. If a selection is not chosen by the ST author, then the selection-based SFRs associated with that selection are not included in the ST;

c) there are no additional security requirements (SFRs or SARs) that are included in the ST that are not also present in the PP/PP-Configuration;

d) in the case where exact conformance is being claimed to multiple PPs, the evaluator determines there are no additional security requirements included in the ST that are not in at least one of the PPs, and that all of the requirements (with the allowances described above) in all of the PPs have been included in the ST;

e) In the case where multiple PPs, or multiple components of a PP-Configuration, to which an ST claims conformance have requirements that need to be combined or de-conflicted, such de-confliction guidance will be provided in those PPs/components as a result of the allow-

[3] In practice, this refers to the union of SFRs defined in the PP-Configuration components.

with analysis performed by the PP/component authors. It is acceptable to include such modified requirements as explicitly allowed in the ST.

The ST includes any optional requirements (and associated SPD elements) from the PP/PP-Configuration as follows:

a) elective requirements defined in the PP/PP-Configuration that the ST wishes to claim; and/or

b) conditional requirements defined in the PP/PP-Configuration that the ST is required to claim (as stipulated in the PP/PP-Configuration) due to the TOE implementation.

If strict conformance is required by the PP to which conformance is being claimed, no conformance claim rationale is required. Instead, the evaluator determines whether the statement of security requirements in the ST is a superset of, or identical to, the statement of security requirements in the PP to which conformance is being claimed (for strict conformance).

If demonstrable conformance is required by the PPs/PP-Configuration to which conformance is being claimed, the evaluator examines the conformance claim rationale to determine that it demonstrates that the statement of security requirements of the ST is equivalent or more restrictive than the statement of security requirements in the PPs/PP-Configuration to which conformance is being claimed.

For:

- SFRs: the conformance rationale in the ST shall demonstrate that the overall set of requirements defined by the SFRs in the ST is equivalent (or more restrictive) than the overall set of requirements defined by the SFRs in the PPs/PP-Configuration. This means that all TOEs that would meet the requirements defined by the set of all SFRs in the ST would also meet the requirements defined by the set of all SFRs in the PPs/PP-Configuration;

- SARs: the ST shall contain all SARs in the PPs/PP-Configuration, but may claim additional SARs or replace SARs by hierarchically stronger SARs. The completion of operations in the ST must be consistent with that in the PPs/PP-Configuration; either the same completion will be used in the ST as that in the PPs/PP-Configuration or a completion that makes the SAR more restrictive (the rules of refinement apply).

For a composed TOE, the evaluator will consider whether the security requirements of the composed TOE are consistent with that specified in the STs for the component TOEs. This is determined in terms of demonstrable conformance. In particular, the evaluator examines the conformance rationale to determine that:

a) the statement of security requirements in the dependent TOE ST relevant to any IT in the operational environment is consistent with the statement of security requirements for the TOE in the base TOE ST. It is not expected that the statement of security requirements for the environment within in the dependent TOE ST will cover all aspects of the statement of security requirements for the TOE in the base TOE ST, as some SFRs may need to be added to the statement of security requirements in the composed TOE ST. However, the statement of security requirements in the base should support the operation of the dependent component;

b) the statement of security objectives in the dependent TOE ST relevant to any IT in the operational environment is consistent with the statement of security requirements for the TOE in the base TOE ST. It is not expected that the statement of security objectives for the

environment within in the dependent TOE ST will cover all aspects of the statement of security requirements for the TOE in the base TOE ST;

c) the statement of security requirements in the composed is consistent with the statements of security requirements in the STs for the component TOEs.

CC Part 3 ASE_CCL.1.12C: *The conformance claim for PP(s) or a PP-Configuration shall be exact, strict, or demonstrable or a list of conformance types.*

12.4.1.3.21 Work unit ASE_CCL.1-20

The evaluator ***shall check*** that the conformance claim for PPs or a PP-Configuration is one of exact, strict, or demonstrable or a list of conformance types.

If the ST claims exact conformance to one PP, then it must claim exact conformance to all PPs, and all PPs must require exact conformance.

CC Part 3 ASE_CCL.1.13C: *If the conformance claim identifies a set of Evaluation Methods and Evaluation Activities derived from CEM work units that shall be used to evaluate the TOE then this set shall include all those that are included in any package, PP, or PP-Module in a PP-Configuration to which the ST claims conformance, and no others.*

12.4.1.3.22 Work unit ASE_CCL.1-21

The evaluator ***shall check*** the conformance claim of the ST to determine that:

a) if any derived Evaluation Methods and Evaluation Activities are required by other items to which the ST claims conformance then these are all identified in the ST, otherwise, no Evaluation Methods and Evaluation Activities are identified;

b) the list of derived Evaluation Methods and Evaluation Activities is sufficiently structured and detailed to unambiguously identify and locate every member of the list;

c) if there is any overlap in the scope of the identified Evaluation Methods and Evaluation Activities (i.e. where an overlay exists as described in CC Part 4) then the rationale for the resulting set of Evaluation Methods and Evaluation Activities is applicable to the TOE as described by the ST.

The intention of this work unit is to ensure that the correct Evaluation Methods and Evaluation Activities can be used when evaluating the TOE described by the ST. This means that identification in the ST need not list individual Evaluation Activities where these are unambiguously included in a listed Evaluation Method. Similarly, where multiple Evaluation Methods or Evaluation Activities are included in a single document then it is sufficient to reference the document, provided this does lead to unambiguous identification of the Evaluation Methods and Evaluation Activities that apply to the ST.

EXAMPLE If a document lists multiple different Evaluation Methods applicable to different use cases then it would not be sufficient to reference the document: the relevant use cases would also have to be identified.

12.5 Security problem definition (ASE_SPD)

12.5.1 Evaluation of sub-activity (ASE_SPD.1)

12.5.1.1 Objectives

The objective of this sub-activity is to determine that the security problem intended to be addressed by the TOE and its operational environment is clearly defined.

12.5.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.5.1.3 Action ASE_SPD.1.1E

12.5.1.3.1 General

CC Part 3 ASE_SPD.1.1C: *The security problem definition shall describe the threats.*

12.5.1.3.2 Work Unit ASE_SPD.1-1

The evaluator ***shall check*** that the security problem definition describes the threats.

If all security objectives are derived from assumptions and/or OSPs 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.

The evaluator determines that the security problem definition describes the threats that must be countered by the TOE and/or operational environment.

CC Part 3 ASE_SPD.1.2C: *All threats shall be described in terms of a threat agent, an asset, and an adverse action.*

12.5.1.3.3 Work unit ASE_SPD.1-2

The evaluator ***shall examine*** the security problem definition to determine that all threats are described in terms of a threat agent, an asset, and an adverse action.

If all security objectives are derived from assumptions and/or OSPs 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.

Threat agents may be further described by aspects such as expertise, resource, opportunity, and motivation.

CC Part 3 ASE_SPD.1.3C: *The security problem definition shall describe the OSPs.*

12.5.1.3.4 Work unit ASE_SPD.1-3

The evaluator ***shall examine*** that the security problem definition describes the OSPs.

If all security objectives are derived from assumptions and threats only, OSPs need not be present in the ST. In this case, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that OSP statements are made in terms of rules or guidelines that must be followed by the TOE and/or its operational environment.

The evaluator determines that each OSP 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.

CC Part 3 ASE_SPD.1.4C: *The security problem definition shall describe the assumptions about the operational environment of the TOE.*

12.5.1.3.5 Work unit ASE_SPD.1-4

The evaluator ***shall examine*** the security problem definition to determine that it describes the assumptions about the operational environment of the TOE.

If there are no assumptions, this work unit is not applicable and is therefore considered to be satisfied.

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

12.6 Security objectives (ASE_OBJ)

12.6.1 Evaluation of sub-activity (ASE_OBJ.1)

12.6.1.1 Objectives

The objective of this sub-activity is to determine whether the security objectives for the operational environment are clearly defined.

12.6.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.6.1.3 Action ASE_OBJ.1.1E

12.6.1.3.1 General

CC Part 3 ASE_OBJ.1.1C: *The statement of security objectives shall describe the security objectives for the operational environment.*

12.6.1.3.2 Work unit ASE_OBJ.1-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the operational environment.

The evaluator checks that the security objectives for the operational environment are identified.

CC Part 3 ASE_OBJ.1.2C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

12.6.1.3.3 Work unit ASE_OBJ.1-2

The evaluator ***shall check*** that the security objectives rationale traces all security objectives for the operational environment back to threats countered by the objectives, OSPs enforced by the objectives and/or assumptions upheld by the objectives.

Each security objective for the operational environment may trace back to threats, OSPs or assumptions, or a combination of threats, OSPs and assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

CC Part 3 ASE_OBJ.1.3C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

12.6.1.3.4 Work unit ASE_OBJ.1-3

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that the tracings from security objectives for the operational 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 operational environment is merely a restatement of an assumption, a justification is required, but this justification may be as minimal as "Security Objective X directly upholds Assumption Y".

12.6.2 Evaluation of sub-activity (ASE_OBJ.2)

12.6.2.1 Objectives

The objective of this sub-activity is to determine whether the security objectives adequately and completely address the security problem definition and that the division of this problem between the TOE and its operational environment is clearly defined.

12.6.2.2 Input

The evaluation evidence for this sub-activity is the ST.

12.6.2.3 Action ASE_OBJ.2.1E

12.6.2.3.1 General

CC Part 3 ASE_OBJ.2.1C: *The statement of security objectives shall describe the security objectives for the TOE and the security objectives for the operational environment.*

12.6.2.3.2 Work unit ASE_OBJ.2-1

The evaluator ***shall check*** that the statement of security objectives defines the security objectives for the TOE and the security objectives for the operational environment.

The evaluator checks that both categories of security objectives are clearly identified and separated from the other category.

CC Part 3 ASE_OBJ.2.2C: *The security objectives rationale shall trace each security objective for the TOE back to threats countered by that security objective and OSPs enforced by that security objective.*

12.6.2.3.3 Work unit ASE_OBJ.2-2

The evaluator ***shall check*** that the security objectives rationale traces all security objectives for the TOE back to threats countered by the objectives and/or OSPs enforced by the objectives.

Each security objective for the TOE may trace back to threats or OSPs, or a combination of threats and OSPs, but it must trace back to at least one threat or OSP.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the TOE has no useful purpose.

CC Part 3 ASE_OBJ.2.3C: *The security objectives rationale shall trace each security objective for the operational environment back to threats countered by that security objective, OSPs enforced by that security objective, and assumptions upheld by that security objective.*

12.6.2.3.4 Work unit ASE_OBJ.2-3

The evaluator ***shall check*** that the security objectives rationale traces the security objectives for the operational environment back to threats countered by that security objective, to OSPs enforced by that security objective, and to assumptions upheld by that security objective.

Each security objective for the operational environment may trace back to threats, OSPs, assumptions, or a combination of threats, OSPs and/or assumptions, but it must trace back to at least one threat, OSP or assumption.

Failure to trace implies that either the security objectives rationale is incomplete, the security problem definition is incomplete, or the security objective for the operational environment has no useful purpose.

CC Part 3 ASE_OBJ.2.4C: *The security objectives rationale shall demonstrate that the security objectives counter all threats.*

12.6.2.3.5 Work unit ASE_OBJ.2-4

The evaluator ***shall examine*** the security objectives rationale to determine that it justifies for each threat that the security objectives are suitable to counter that threat.

If no security objectives trace back to the threat, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

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

Note that the tracings from security objectives to threats provided in the security objectives rationale may be 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 may be as minimal as “Security Objective X directly counters Threat Y”.

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

CC Part 3 ASE_OBJ.2.5C: *The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.*

12.6.2.3.6 Work unit ASE_OBJ.2-5

The evaluator ***shall examine*** the security objectives rationale to determine that for each OSP it justifies that the security objectives are suitable to enforce that OSP.

If no security objectives trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all security objectives that trace back to that OSP are achieved, the OSP is enforced.

The evaluator also determines that each security objective that traces back to an OSP is necessary: when the security objective is achieved it actually contributes to the enforcement of the OSP.

Note that the tracings from security objectives to OSPs provided in the security objectives rationale may be part of a justification, but do not constitute a justification by themselves. In the case that a security objective is merely a statement reflecting the intent to enforce a particular OSP, a justification is required, but this justification may be as minimal as "Security Objective X directly enforces OSP Y".

CC Part 3 ASE_OBJ.2.6C: *The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.*

12.6.2.3.7 Work unit ASE_OBJ.2-6

The evaluator ***shall examine*** the security objectives rationale to determine that for each assumption for the operational environment it contains an appropriate justification that the security objectives for the operational environment are suitable to uphold that assumption.

If no security objectives for the operational environment trace back to the assumption, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an assumption about the operational environment of the TOE demonstrates that the security objectives are sufficient: if all security objectives for the operational environment that trace back to that assumption are achieved, the operational environment upholds the assumption.

The evaluator also determines that each security objective for the operational environment that traces back to an assumption about the operational environment of the TOE is necessary: when the security objective is achieved it actually contributes to the operational environment upholding the assumption.

Note that the tracings from security objectives for the operational 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 operational environment is merely a restatement of an assumption, a justification is required, but this justification may be as minimal as “Security Objective X directly upholds Assumption Y”.

12.7 Extended components definition (ASE_ECD)

12.7.1 Evaluation of sub-activity (ASE_ECD.1)

12.7.1.1 Objectives

The objective of this sub-activity is to determine whether extended components have been clearly and unambiguously defined, and whether they are necessary, i.e. they may not be clearly expressed using existing CC Part 2 or CC Part 3 components.

12.7.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.7.1.3 Action ASE_ECD.1.1E

12.7.1.3.1 General

CC Part 3 ASE_ECD.1.1C: *The statement of security requirements shall identify all extended security requirements.*

12.7.1.3.2 Work unit ASE_ECD.1-1

The evaluator ***shall check*** that all security requirements in the statement of security requirements that are not identified as extended requirements are present in CC Part 2 or in CC Part 3.

CC Part 3 ASE_ECD.1.2C: *The extended components definition shall define an extended component for each extended security requirement.*

12.7.1.3.3 Work unit ASE_ECD.1-2

The evaluator ***shall check*** that the extended components definition defines an extended component for each extended security requirement.

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

If the extended component definition(s) are taken from a PP/PP-Configuration that has previously been certified then the PP/PP-Configuration evaluation results may be reused as described in 12.2.1 (Re-using the evaluation results of certified PPs).

A single extended component may be used to define multiple iterations of an extended security requirement; it is not necessary to repeat this definition for each iteration.

CC Part 3 ASE_ECD.1.3C: *The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.*

12.7.1.3.4 Work unit ASE_ECD.1-3

The evaluator ***shall examine*** the extended components definition to determine that it describes how each extended component fits into the existing CC components, families, and classes.

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

The evaluator determines that each extended component is either:

a) a member of an existing CC Part 2 or CC Part 3 family; or

b) a member of a new family defined in the ST.

If the extended component is a member of an existing CC Part 2 or CC Part 3 family, the evaluator determines that the extended components definition adequately describes why the extended component should be a member of that family and how it relates to other components of that family.

If the extended component is a member of a new family defined in the ST, the evaluator confirms that the extended component is not appropriate for an existing family.

If the ST defines new families, the evaluator determines that each new family is either:

a) a member of an existing CC Part 2 or CC Part 3 class; or

b) a member of a new class defined in the ST.

If the family is a member of an existing CC Part 2 or CC Part 3 class, the evaluator determines that the extended components definition adequately describes why the family should be a member of that class and how it relates to other families in that class.

If the family is a member of a new class defined in the ST, the evaluator confirms that the family is not appropriate for an existing class.

12.7.1.3.5 Work unit ASE_ECD.1-4

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended component identifies all applicable dependencies of that component.

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

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

CC Part 3 ASE_ECD.1.4C: *The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.*

12.7.1.3.6 Work unit ASE_ECD.1-5

The evaluator ***shall examine*** the extended components definition to determine that each extended functional component uses the existing CC Part 2 components as a model for presentation.

If the ST does not contain extended SFRs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended functional component is consistent with CC Part 2, 6.1.4, Component structure.

If the extended functional component uses operations, the evaluator determines that the extended functional component is consistent with CC Part 1, 8.2, Operations.

If the extended functional component is hierarchical to an existing functional component, the evaluator determines that the extended functional component is consistent with CC Part 2, 6.2.2, Component changes highlighting.

12.7.1.3.7 Work unit ASE_ECD.1-6

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional family uses the existing CC functional families as a model for presentation.

If the ST does not define new functional families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional families are defined consistent with CC Part 2, 6.1.3, Family structure.

12.7.1.3.8 Work unit ASE_ECD.1-7

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new functional class uses the existing CC functional classes as a model for presentation.

If the ST does not define new functional classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new functional classes are defined consistent with CC Part 2, 6.1.2, Class structure.

12.7.1.3.9 Work unit ASE_ECD.1-8

The evaluator ***shall examine*** the extended components definition to determine that each definition of an extended assurance component uses the existing CC Part 3 components as a model for presentation.

If the ST does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that the extended assurance component definition is consistent with CC Part 3, 6.2, Assurance component structure.

If the extended assurance component uses operations, the evaluator determines that the extended assurance component is consistent with CC Part 1, 8.2, Operations.

If the extended assurance component is hierarchical to an existing assurance component, the evaluator determines that the extended assurance component is consistent with CC Part 3, 6.1.2, Assurance component structure.

12.7.1.3.10 Work unit ASE_ECD.1-9

The evaluator ***shall examine*** the extended components definition to determine that, for each defined extended assurance component, applicable methodology has been provided.

If the ST does not contain extended SARs, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that, for each evaluator action element of each extended SAR, one or more work units are provided and that successfully performing all work units for a given evaluator action element will demonstrate that the element has been achieved.

12.7.1.3.11 Work unit ASE_ECD.1-10

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance family uses the existing CC assurance families as a model for presentation.

If the ST does not define new assurance families, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance families are defined consistent with CC Part 3, 6.1.2, Assurance family structure.

12.7.1.3.12 Work unit ASE_ECD.1-11

The evaluator ***shall examine*** the extended components definition to determine that each definition of a new assurance class uses the existing CC assurance classes as a model for presentation.

If the ST does not define new assurance classes, this work unit is not applicable and therefore considered to be satisfied.

The evaluator determines that all new assurance classes are defined consistent with CC Part 3, 6.1.1, Assurance class structure.

12.7.1.3.13 General

CC Part 3 ASE_ECD.1.5C: *The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements may be demonstrated.*

12.7.1.3.14 Work unit ASE_ECD.1-12

The evaluator ***shall examine*** the extended components definition to determine that each element in each extended component is measurable and states objective evaluation requirements, such that conformance or nonconformance can be demonstrated.

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

The evaluator determines that elements of extended functional components are stated in such a way that they are testable, and traceable through the appropriate TSF representations.

The evaluator also determines that elements of extended assurance components avoid the need for subjective evaluator judgement.

The evaluator is reminded that whilst being measurable and objective is appropriate for all evaluation criteria, it is acknowledged that no formal method exists to prove such properties. Therefore, the existing CC functional and assurance components are to be used as a model for determining what constitutes conformance with this requirement.

12.7.1.3.15 Action ASE_ECD.1.2E

12.7.1.3.16 Work unit ASE_ECD.1-13

The evaluator ***shall examine*** the extended components definition to determine that each extended component cannot be clearly expressed using existing components.

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

The evaluator should take components from CC Part 2 and CC Part 3, other extended components that have been defined in the ST, combinations of these components, and possible operations on these components into account when making this determination.

The evaluator is reminded that the role of this work unit is to preclude unnecessary duplication of components, that is, components that may be clearly expressed by using other components. The evaluator should not undertake an exhaustive search of all possible combinations of components including operations in an attempt to find a way to express the extended component by using existing components.

12.8 Security requirements (ASE_REQ)

12.8.1 Evaluation of sub-activity (ASE_REQ.1)

12.8.1.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs counter the threats and implement the organisational security policies of the TOE.

12.8.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.8.1.3 Action ASE_REQ.1.1E

12.8.1.3.1 General

CC Part 3 ASE_REQ.1.1C: *The statement of security requirements shall describe the SFRs and the SARs.*

12.8.1.3.2 Work unit ASE_REQ.1-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2;

b) by reference to an extended component in the extended components definition of the ST;

c) by reference to a PP that the ST claims to be conformant with, including any optional requirements defined in the PP;

d) by reference to a security requirements package that the ST claims to be conformant with;

e) by reproduction in the ST.

It is not required to use the same means of identification for all SFRs.

12.8.1.3.3 Work unit ASE_REQ.1-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that each SAR is identified by one of the following means:

a) by reference to an individual component in CC Part 3;

b) by reference to an extended component in the extended components definition of the ST;

c) by reference to a PP that the ST claims to be conformant with;

d) by reference to a security requirements package that the ST claims to be conformant with;

e) by reproduction in the ST.

It is not required to use the same means of identification for all SARs.

Note that if optional requirements are defined by the PP, there may be associated threats that are covered by this work unit.

CC Part 3 ASE_REQ.1.2C: *For a single-assurance ST, the statement of security requirements shall define the global set of SARs that apply to the entire TOE. The sets of SARs shall be consistent with the PPs or PP-Configuration to which the ST claims conformance.*

12.8.1.3.4 Work unit ASE_REQ.1-3

The evaluator ***shall check*** that the statement of security requirements defines the global set of SARs that apply to the entire TOE.

The evaluator ***shall examine*** the sets of SARs to determine that they are consistent with the PPs or the PP-Configuration to which the ST claims conformance.

CC Part 3 ASE_REQ.1.3C: *For a multi-assurance ST, the statement of security requirements shall define the global set of SARs that apply to the entire TOE and the sets of SARs that apply to each sub-TSF. The sets of SARs shall be consistent with the multi-assurance PP-Configuration to which the ST claims conformance.*

12.8.1.3.5 Work unit ASE_REQ.1-4

For a multi-assurance ST, the evaluator ***shall examine*** the statement of security requirements to determine that it defines the global set of SARs that apply to the entire TOE.

The TSF organization associates sub-TSFs with specific assurance requirements. It may happen that sub-TSFs are implemented by different sets of subsystems/modules, but there may also be some degree of overlap: a subsystem or module may implement functionalities belonging to two different sub-TSFs. This means that the two sets of SARs apply to the common subsystem or module (i.e. the union of the sets of SARs applies). In both cases, for each sub-TSF, all of the other sub-TSFs belong to the TOE and the corresponding subsystems/modules must be evaluated through the prism of the requirements of the sub-TSF.

12.8.1.3.6 Work unit ASE_REQ.1-5

The evaluator ***shall check*** that the SARs that apply to each sub-TSF are defined in the ST and that they are either identical to the ones defined in the PP-Configuration or augmented. Augmentation is only allowed when strict or demonstrable conformance is being claimed by the ST to the PP-Configuration.

12.8.1.3.7 Work unit ASE_REQ.1-6

For augmented sets of SARs, the evaluator ***shall check*** that a rationale is provided.

12.8.1.3.8 Work unit ASE_REQ.1-7

The evaluator ***shall check*** that the sets of SARs defined in the ST are consistent with the multi-assurance PP-Configuration to which the ST claims conformance.

CC Part 3 ASE_REQ.1.4C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

12.8.1.3.9 Work unit ASE_REQ.1-8

The evaluator ***shall examine*** the ST to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the ST defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the ST author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This may be especially applicable if the same terms are used in the rest of the ST.

CC Part 3 ASE_REQ.1.5C: *The statement of security requirements shall identify all operations on the security requirements.*

12.8.1.3.10 Work unit ASE_REQ.1-9

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

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

CC Part 3 ASE_REQ.1.6C: *All operations shall be performed correctly.*

12.8.1.3.11 Work unit ASE_REQ.1-10

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.1.3.12 Work unit ASE_REQ.1-11

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.1.3.13 Work unit ASE_REQ.1-12

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.1.3.14 Work unit ASE_REQ.1-13

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

CC Part 3 ASE_REQ.1.7C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

12.8.1.3.15 Work unit ASE_REQ.1-14

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that a security requirements rationale is provided which justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

a) why the dependency is not necessary or useful, in which case no further information is required; or

b) that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

CC Part 3 ASE_REQ.1.8C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) counter all threats for the TOE.*

12.8.1.3.16 Work unit ASE_REQ.1-15

The evaluator ***shall examine*** the security requirements rationale to determine that for each threat it demonstrates that the SFRs are suitable to meet that threat and that each SFR is traced back to at least one threat (or OSP) for the TOE.

If no SFRs trace back to a threat, the evaluator action related to this work unit is assigned a fail verdict as it implies that either the security requirements rationale is incomplete, the security objectives for the TOE are incomplete, or some SFRs have no useful purpose.

The evaluator determines that the justification for a threat shows whether the threat is removed, diminished or mitigated.

The evaluator determines that the justification for a threat demonstrates that the SFRs are sufficient: if all SFRs that trace back to the threat are achieved then, in the context of any applicable OSPs and assumptions, the threat is removed, sufficiently diminished, or the effects of the threat are sufficiently mitigated.

Note that simply listing in the security requirements rationale the SFRs associated with each threat may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly counters Threat Y".

The evaluator also determines that each SFR that traces back to a threat is necessary: when the SFR is implemented it actually contributes to the removal, diminishing or mitigation of that threat.

CC Part 3 ASE_REQ.1.9C: *The security requirements rationale shall demonstrate that the SFRs (in conjunction with the security objectives for the environment) enforce all OSPs.*

12.8.1.3.17 Work unit ASE_REQ.1-16

The evaluator ***shall examine*** the security requirements rationale to determine that for each OSP it justifies that the SFRs are suitable to enforce that OSP.

If no SFRs or security objectives for the operational environment trace back to the OSP, the evaluator action related to this work unit is assigned a fail verdict.

The evaluator determines that the justification for an OSP demonstrates that the security objectives are sufficient: if all SFRs that trace back to that OSP are achieved then, in the context of any applicable assumptions, the OSP is enforced.

The evaluator also determines that each SFR that traces back to an OSP is necessary: when the SFR is implemented it actually contributes to the enforcement of the OSP.

Note that simply listing in the security requirements rationale the SFRs associated with each OSP may be part of a justification, but does not constitute a justification by itself. A descriptive justification is required, although in simple cases this justification may be as minimal as "SFR X directly enforces OSP Y".

CC Part 3 ASE_REQ.1.10C: *The security requirements rationale shall explain why the SARs were chosen.*

12.8.1.3.18 Work unit ASE_REQ.1-17

The evaluator ***shall examine*** the security requirements rationale to determine that the inclusion of each chosen SAR has been explained and is justified.

CC Part 3 ASE_REQ.1.11C: *The statement of security requirements shall be internally consistent.*

12.8.1.3.19 Work unit ASE_REQ.1-18

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

a) all optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

1) All optional requirements marked as conditional in a PP or PP-Configuration Component to which the ST claims conformance are included if the TOE implements functionality that is covered by the requirement. Elective optional requirements may be omitted;

b) all optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to "all objects", "all subjects" etc., that these requirements do not conflict.

Some possible conflicts are:

a) an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended SAR specifying an open source review;

b) FAU_GEN.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

c) FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

d) multiple iterations of FDP_ACC.1 Subset access control especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

CC Part 3 ASE_REQ.1.12C: *If the ST defines sets of SARs that expand the sets of SARs of the PPs or the PP-Configuration it claims conformance to, the security requirements rationale shall include an assurance rationale that justifies the consistency of the extension and provides a rationale for the disposition of any Evaluation Methods/Evaluation Activities identified in the conformance statement that are affected by the extension of the sets of SARs.*

12.8.1.3.20 Work unit ASE_REQ.1-19

If an ST extends the sets of SARs of the PP(s) or PP-Configuration it claims conformance to, the evaluator ***shall examine*** the security requirements rationale to determine that it includes an assurance rationale justifying the consistency of the extension with regard to the SPD, together with a rationale for the disposition of any Evaluation Methods/Evaluation Activities defined in the PP(s) or PP-Configuration to which it claims conformance that are affected by the extension of the sets of SARs.

If the ST does not claim conformance to one or more PPs, or to a PP-Configuration, this work unit is not applicable and should be considered satisfied.

If the ST claims exact conformance to one or more PPs, or to a PP-Configuration, then augmentation of the SARs is not allowed and the evaluator verifies that either no augmentation has been performed (in which case this work unit is considered satisfied), or that the conformance claim is either strict or demonstrable."

Where the expansion of the SARs affects any Evaluation Methods/Evaluation Activities defined in the PP or PP-Configuration to which the ST claims conformance, the evaluator examines the assurance rationale to confirm that it preserves the relevant aspects of the affected Evaluation Methods/Evaluation Activities, taking into account the context in which the ST is to be used. This includes a check that the assurance resulting from meeting the expanded SARs will be as least as strong as that generated by the original Evaluation Methods/Evaluation Activities.

If either rationale is missing, or incomplete, then the evaluator action related to this work unit is assigned a fail verdict.

12.8.2 Evaluation of sub-activity (ASE_REQ.2)

12.8.2.1 Objectives

The objective of this sub-activity is to determine whether the SFRs and SARs are clear, unambiguous and well-defined, whether they are internally consistent, and whether the SFRs meet the security objectives of the TOE.

12.8.2.2 Input

The evaluation evidence for this sub-activity is the ST.

12.8.2.3 Action ASE_REQ.2.1E

12.8.2.3.1 General

CC Part 3 ASE_REQ.2.1C: *The statement of security requirements shall describe the SFRs and the SARs.*

12.8.2.3.2 Work unit ASE_REQ.2-1

The evaluator ***shall check*** that the statement of security requirements describes the SFRs.

The evaluator determines that each SFR is identified by one of the following means:

a) by reference to an individual component in CC Part 2;

b) by reference to an extended component in the extended components definition of the ST;

c) by reference to an individual component in a PP that the ST claims to be conformant with, including any optional requirements defined in the PP;

d) by reference to an individual component in a security requirements package that the ST claims to be conformant with;

e) by reproduction in the ST.

It is not required to use the same means of identification for all SFRs.

12.8.2.3.3 Work unit ASE_REQ.2-2

The evaluator ***shall check*** that the statement of security requirements describes the SARs.

The evaluator determines that all SARs are identified by one of the following means:

a) by reference to an individual component in CC Part 3

b) by reference to an extended component in the extended components definition of the ST;

c) by reference to an individual component in a PP that the ST claims to be conformant with;

d) by reference to an individual component in a security requirements package that the ST claims to be conformant with;

e) by reproduction in the ST.

It is not required to use the same means of identification for all SARs.

Note that if optional requirements are defined by the PP, there may be associated threats that are covered by this work unit.

CC Part 3 ASE_REQ.2.2C: *For a single-assurance ST, the statement of security requirements shall define the global set of SARs that apply to the entire TOE. The sets of SARs shall be consistent with the PPs or PP-Configuration to which the ST claims conformance.*

12.8.2.3.4 Work unit ASE_REQ.2-3

The evaluator ***shall check*** that the statement of security requirements defines the global set of SARs that apply to the entire TOE.

12.8.2.3.5 Work unit ASE_REQ.2-4

The evaluator ***shall examine*** the sets of SARs to determine that they are consistent with the PPs or PP-Configuration to which the ST claims conformance.

CC Part 3 ASE_REQ.2.3C: *For a multi-assurance ST, the statement of security requirements shall define the global set of SARs that apply to the entire TOE and the sets of SARs that apply to each sub-TSF. The sets of SARs shall be consistent with the multi-assurance PP-Configuration to which the ST claims conformance.*

12.8.2.3.6 Work unit ASE_REQ.2-5

For a multi-assurance ST, the evaluator ***shall examine*** the statement of security requirements to determine that it defines the global set of SARs that apply to the entire TOE.

The TSF organization associates sub-TSFs with specific assurance requirements. It may happen that sub-TSFs are implemented by different sets of subsystems/modules, but there may also be some degree of overlap: a subsystem or module may implement functionalities belonging to two different sub-TSFs. This means that the two sets of SARs apply to the common subsystem or module (i.e. the union of the sets of SARs applies). In both cases, for each sub-TSF, all of the other sub-TSFs belong to the TOE and the corresponding subsystems/modules must be evaluated through the prism of the requirements of the sub-TSF.

12.8.2.3.7 Work unit ASE_REQ.2-6

The evaluator ***shall check*** that the SARs that apply to each sub-TSF are defined in the ST and that they are either identical to the ones defined in the PP-Configuration or augmented. Augmentation is only allowed when strict or demonstrable conformance is being claimed by the ST to the PP-Configuration.

12.8.2.3.8 Work unit ASE_REQ.2-7

For augmented sets of SARs, the evaluator ***shall check*** that a rationale is provided.

12.8.2.3.9 Work unit ASE_REQ.2-8

The evaluator ***shall check*** that the sets of SARs defined in the ST are consistent with the multi-assurance PP-Configuration to which the ST claims conformance.

CC Part 3 ASE_REQ.2.4C: *All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.*

12.8.2.3.10 Work unit ASE_REQ.2-9

The evaluator ***shall examine*** the ST to determine that all subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs are defined.

The evaluator determines that the ST defines all:

- (types of) subjects and objects that are used in the SFRs;
- (types of) security attributes of subjects, users, objects, information, sessions and/or resources, possible values that these attributes may take and any relations between these values (e.g. top_secret is "higher" than secret);
- (types of) operations that are used in the SFRs, including the effects of these operations;
- (types of) external entities in the SFRs;
- other terms that are introduced in the SFRs and/or SARs by completing operations, if these terms are not immediately clear, or are used outside their dictionary definition.

The goal of this work unit is to ensure that the SFRs and SARs are well-defined and that no misunderstanding may occur due to the introduction of vague terms. This work unit should not be taken into extremes, by forcing the ST author to define every single word. The general audience of a set of security requirements should be assumed to have a reasonable knowledge of IT, security and the CC.

All of the above may be presented in groups, classes, roles, types or other groupings or characterisations that allow easy understanding.

The evaluator is reminded that these lists and definitions do not have to be part of the statement of security requirements, but may be placed (in part or in whole) in different subclauses. This may be especially applicable if the same terms are used in the rest of the ST.

CC Part 3 ASE_REQ.2.5C: *The statement of security requirements shall identify all operations on the security requirements.*

12.8.2.3.11 Work unit ASE_REQ.2-10

The evaluator ***shall check*** that the statement of security requirements identifies all operations on the security requirements.

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

CC Part 3 ASE_REQ.2.6C: *All operations shall be performed correctly.*

12.8.2.3.12 Work unit ASE_REQ.2-11

The evaluator ***shall examine*** the statement of security requirements to determine that all assignment operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.2.3.13 Work unit ASE_REQ.2-12

The evaluator ***shall examine*** the statement of security requirements to determine that all iteration operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.2.3.14 Work unit ASE_REQ.2-13

The evaluator ***shall examine*** the statement of security requirements to determine that all selection operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

12.8.2.3.15 Work unit ASE_REQ.2-14

The evaluator ***shall examine*** the statement of security requirements to determine that all refinement operations are performed correctly.

Guidance on the correct performance of operations may be found in CC Part 1, 8.2 Operations.

CC Part 3 ASE_REQ.2.7C: *Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.*

12.8.2.3.16 Work unit ASE_REQ.2-15

The evaluator ***shall examine*** the statement of security requirements to determine that each dependency of the security requirements is either satisfied, or that the security requirements rationale justifies the dependency not being satisfied.

A dependency is satisfied by the inclusion of the relevant component (or one that is hierarchical to it) within the statement of security requirements. The component used to satisfy the dependency should, if necessary, be modified by operations to ensure that it actually satisfies that dependency.

A justification that a dependency is not met should address either:

a) why the dependency is not necessary or useful, in which case no further information is required; or

b) that the dependency has been addressed by the operational environment of the TOE, in which case the justification should describe how the security objectives for the operational environment address this dependency.

CC Part 3 ASE_REQ.2.8C: *The security requirements rationale shall demonstrate that the SFRs meet all security objectives for the TOE.*

12.8.2.3.17 Work unit ASE_REQ.2-16

The evaluator ***shall examine*** the security requirements rationale to determine that for each security objective for the TOE it demonstrates that the SFRs are suitable to meet that security objective for the TOE.

If no SFRs trace back to the security objective for the TOE, the evaluator action related to this work unit is assigned a fail verdict.

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

The evaluator also determines that each SFR that traces back to a security objective for the TOE is necessary: when the SFR is satisfied, it actually contributes to achieving the security objective.

Note that the tracings from SFRs 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.

CC Part 3 ASE_REQ.2.9C: *The security requirements rationale shall explain why the SARs were chosen.*

12.8.2.3.18 Work unit ASE_REQ.2-17

The evaluator ***shall check*** that the security requirements rationale explains why the SARs were chosen.

The evaluator is reminded that any explanation is correct, as long as it is coherent and neither the SARs nor the explanation have obvious inconsistencies with the remainder of the ST.

An example of an obvious inconsistency between the SARs and the remainder of the ST would be to have threat agents that are very capable, but an AVA_VAN SAR that does not protect against these threat agents.

CC Part 3 ASE_REQ.2.10C: *The statement of security requirements shall be internally consistent.*

12.8.2.3.19 Work unit ASE_REQ.2-18

The evaluator ***shall examine*** the statement of security requirements to determine that it is internally consistent.

The evaluator determines that the combined set of all SFRs and SARs is internally consistent. With respect to optional requirements, the evaluator determines that:

a) all optional requirements either trace to an SPD element that is itself not optional, or trace to an SPD element that is clearly associated with that optional SFR;

    1) All optional requirements marked as conditional in a PP or PP-Configuration Component to which the ST claims conformance are included if the TOE implements functionality that is covered by the requirement. Elective optional requirements may be omitted;

b) all optional requirements do not conflict with non-optional requirements (a capability cannot be both required and optional; however, a base capability can be required with enhancements to that capability being specified as optional).

The evaluator determines that on all occasions where different security requirements apply to the same types of developer evidence, events, operations, data, tests to be performed etc. or to “all objects”, “all subjects” etc., that these requirements do not conflict.

Some possible conflicts are:

a) an extended SAR specifying that the design of a certain cryptographic algorithm is to be kept secret, and another extended assurance requirement specifying an open source review;

b) FAU_GEN.1 Audit data generation specifying that subject identity is to be logged, FDP_ACC.1 Subset access control specifying who has access to these logs, and FPR_UNO.1 Unobservability specifying that some actions of subjects should be unobservable to other subjects. If the subject that should not be able to see an activity may access logs of this activity, these SFRs conflict;

c) FDP_RIP.1 Subset residual information protection specifying deletion of information no longer needed, and FDP_ROL.1 Basic rollback specifying that a TOE may return to a previous state. If the information that is needed for the rollback to the previous state has been deleted, these requirements conflict;

d) multiple iterations of FDP_ACC.1 Subset access control especially where some iterations cover the same subjects, objects, or operations. If one access control SFR allows a subject to perform an operation on an object, while another access control SFR does not allow this, these requirements conflict.

CC Part 3 ASE_REQ.2.11C: *If the ST defines sets of SARs that expand the sets of SARs of the PPs or PP-Configuration it claims conformance to, the security requirements rationale shall include an assurance rationale that justifies the consistency of the extension, and provides a rationale for the disposition of any Evaluation Methods and Evaluation Activities identified in the conformance statement that are affected by the extension of the sets of SARs.*

12.8.2.3.20 Work unit ASE_REQ.2-19

If an ST that extends the sets of SARs of the PP-Configuration it claims conformance to, the evaluator ***shall examine*** the security requirements rationale to determine that it includes an assurance rationale justifying the consistency of the extension with regard to the SPD, together with a rationale for the disposition of any Evaluation Methods/Evaluation Activities defined in the PP(s) or PP-Configuration to which it claims conformance that are affected by the extension of the sets of SARs.

If the ST does not claim conformance to one or more PPs, or to a PP-Configuration, this work unit is not applicable and should be considered satisfied.

If the ST claims exact conformance to one or more PPs, or to a PP-Configuration, then augmentation of the SARs is not allowed and the evaluator verifies that either no augmentation has been performed (in which case this work unit is considered satisfied), or that the conformance claim is either strict or demonstrable.

Where the expansion of the SARs affects any Evaluation Methods/Evaluation Activities defined in the PP or PP-Configuration to which the ST claims conformance, the evaluator examines the assurance rationale to confirm that it preserves the relevant aspects of the affected Evaluation Methods/Evaluation Activities, taking into account the context in which the ST is to be used. This includes a check that the assurance resulting from meeting the expanded SARs will be as least as strong as that generated by the original Evaluation Methods/Evaluation Activities.

If either rationale is missing, or incomplete, then the evaluator action related to this work unit is assigned a fail verdict.

12.9 TOE summary specification (ASE_TSS)

12.9.1 Evaluation of sub-activity (ASE_TSS.1)

12.9.1.1 Objectives

The objective of this sub-activity is to determine whether the TOE summary specification addresses all SFRs, and whether the TOE summary specification is consistent with other narrative descriptions of the TOE.

12.9.1.2 Input

The evaluation evidence for this sub-activity is the ST.

12.9.1.3 Action ASE_TSS.1.1E

12.9.1.3.1 General

CC Part 3 ASE_TSS.1.1C: *The TOE summary specification shall describe how the TOE meets each SFR.*

12.9.1.3.2 Work unit ASE_TSS.1-1

The evaluator ***shall examine*** the TOE summary specification to determine that it describes how the TOE meets each SFR.

The evaluator determines that the TOE summary specification provides, for each SFR from the statement of security requirements, a description on how that SFR is met.

The evaluator is reminded that the objective of each description is to provide potential consumers of the TOE with a high-level view of how the developer intends to satisfy each SFR and that the descriptions therefore should not be overly detailed. Often several SFRs will be implemented in one context; for instance, a password authentication mechanism may implement FIA_UAU.1, FIA_SOS.1 and FIA_UID.1. Therefore, usually the TSS will not consist of a long list with texts for each single SFR, but complete groups of SFRs may be covered by one text passage.

For a composed TOE, the evaluator also determines that it is clear which component provides each SFR or how the components combine to meet each SFR.

12.9.1.4 Action ASE_TSS.1.2E

12.9.1.4.1 Work unit ASE_TSS.1-2

The evaluator ***shall examine*** the TOE summary specification to determine that it is consistent with the TOE overview and the TOE description.

The TOE overview, TOE description, and TOE summary specification describe the TOE in a narrative form at increasing levels of detail. These descriptions therefore need to be consistent.

12.9.2 Evaluation of sub-activity (ASE_TSS.2)

12.9.2.1 Objectives

The objective of this sub-activity is to determine whether the TOE summary specification addresses all SFRs, whether the TOE summary specification addresses interference, logical tampering and bypass, and whether the TOE summary specification is consistent with other narrative descriptions of the TOE.

12.9.2.2 Input

The evaluation evidence for this sub-activity is the ST.

12.9.2.3 Action ASE_TSS.2.1E

12.9.2.3.1 General

CC Part 3 ASE_TSS.2.1C: *The TOE summary specification shall describe how the TOE meets each SFR.*

12.9.2.3.2 Work unit ASE_TSS.2-1

The evaluator ***shall examine*** the TOE summary specification to determine that it describes how the TOE meets each SFR.

The evaluator determines that the TOE summary specification provides, for each SFR from the statement of security requirements, a description on how that SFR is met.

The evaluator is reminded that the objective of each description is to provide potential consumers of the TOE with a high-level view of how the developer intends to satisfy each SFR and that the descriptions therefore should not be overly detailed. Often several SFRs will be implemented in one context; for instance, a password authentication mechanism may implement FIA_UAU.1, FIA_SOS.1 and FIA_UID.1. Therefore, usually the TSS will not consist of a long list with texts for each single SFR, but complete groups of SFRs may be covered by one text passage.

For a composed TOE, the evaluator also determines that it is clear which component provides each SFR or how the components combine to meet each SFR.

CC Part 3 ASE_TSS.2.2C: *The TOE summary specification shall describe how the TOE protects itself against interference and logical tampering.*

12.9.2.3.3 Work unit ASE_TSS.2-2

The evaluator ***shall examine*** the TOE summary specification to determine that it describes how the TOE protects itself against interference and logical tampering.

The evaluator is reminded that the objective of each description is to provide potential consumers of the TOE with a high-level view of how the developer intends to provide protection against interference and logical tampering and that the descriptions therefore should not be overly detailed.

For a composed TOE, the evaluator also determines that it is clear which component provides the protection or how the components combine to provide protection.

CC Part 3 ASE_TSS.2.3C: *The TOE summary specification shall describe how the TOE protects itself against bypass.*

12.9.2.3.4 Work unit ASE_TSS.2-3

The evaluator ***shall examine*** the TOE summary specification to determine that it describes how the TOE protects itself against bypass.

The evaluator is reminded that the objective of each description is to provide potential consumers of the TOE with a high-level view of how the developer intends to provide protection against bypass and that the descriptions therefore should not be overly detailed.

For a composed TOE, the evaluator also determines that it is clear which component provides the protection or how the components combine to provide protection.

12.9.2.4 Action ASE_TSS.2.2E

12.9.2.4.1 Work unit ASE_TSS.2-4

The evaluator ***shall examine*** the TOE summary specification to determine that it is consistent with the TOE overview and the TOE description.

The TOE overview, TOE description, and TOE summary specification describe the TOE in a narrative form at increasing levels of detail. These descriptions therefore need to be consistent.

12.10 Consistency of composite product Security Target (ASE_COMP)

12.10.1 General

The composite-specific work units defined here are intended to be integrated as refinements to the evaluation activities of the ASE class listed in the following table. The other activities of the ASE class do not require composite-specific work units.

Table 1 — ASE_COMP

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
ASE_OBJ	ASE_OBJ.2.1E	ASE_OBJ.2-1	ASE_COMP.1-5
	ASE_OBJ.2.1E	ASE_OBJ.2-1	ASE_COMP.1-6
	ASE_OBJ.2.1E	ASE_OBJ.2-3	ASE_COMP.1-6
ASE_REQ	ASE_REQ.1.1E	ASE_REQ.1-16	ASE_COMP.1-1
	ASE_REQ.2.1E	ASE_REQ.2-13	ASE_COMP.1-1
	ASE_REQ.1.1E	ASE_REQ.1-16	ASE_COMP.1-2
	ASE_REQ.2.1E	ASE_REQ.2-13	ASE_COMP.1-2
	ASE_REQ.2.1E	ASE_REQ.2-12	ASE_COMP.1-3
	ASE_REQ.2.1E	ASE_REQ.2-4	ASE_COMP.1-4

12.10.2 Evaluation of sub-activity (ASE_COMP.1)

12.10.2.1 Objectives

The aim of this activity is to determine whether the Security Target of the composite product[4] does not contradict the Security Target of the related base component[5] [6].

[4] Denoted by composite product Security Target or composite-ST in the in the composite specific work units.

[5] Denoted by base component Security Target or base-ST in the in the composite specific work units.

[6] Generally, a Security Target expresses a security policy for the TOE defined.

12.10.2.2 Application notes

A Security Target for the composite product shall be written and evaluated.

The composite product evaluator shall examine that the Security Target of the composite product does not contradict the Security Target of the related base component. In particular, this means that the composite product evaluator shall examine the composite product Security Target and the base component Security Target for any conflicting assumptions, compatibility of security objectives, security requirements and security functionality needed by the dependent component.

The composite product evaluation sponsor shall ensure that the Security Target of the base component is available for the dependent component developer, for the composite product evaluator and for the composite product evaluation authority. The information available in the public version of the base component Security Target may not be sufficient.

These application notes aid the developer to create as well as the evaluator to analyse a composite product Security Target and describe a general methodology for it. For detailed information / guidance please refer to the single work units below.

In order to create a composite product Security Target the developer should perform the following steps:

Step 1: The developer formulates a preliminary Security Target for the composite product (the composite-ST) using the standard code of practice. The composite-ST can be formulated independently of the Security Target of the composite product's related base component (the base-ST) – at least as long as there are no formal PP conformance claims.

Step 2: The developer determines the overlap between the base-ST and the composite-ST by analysing and comparing their respective TOE Security Functionality (TSF)[7] [8].

Step 3: The developer determines under which conditions they can trust in and rely on the base component-TSF being used by the composite-ST without a new examination.

Having undertaken these steps the developer completes the preliminary Security Target for the composite product.

It is not mandatory that the composite product and its related base component are being evaluated according to the same edition of the CC (all parts). It is due to the fact that the dependent component of the composite product can rely on some security services of the base component, if (i) the assurance level of the base component covers the intended assurance level of the composite product and (ii) the base component evaluation is valid (i.e. accepted by the base component evaluation authority) and up-to-date. Equivalence of single assurance components (and, hence, of assurance levels) belonging to different CC editions shall be established / acknowledged by the composite product evaluation authority.

If conformance to a PP is claimed, e.g. a composite product Security Target claims conformance to a PP (that possibly claims conformance to a further PP), the consistency check can be reduced

---

[7] Because the TSF enforce the Security Target (together with the organisational measures enforcing the security objectives for the operational environment of the TOE).

[8] The comparison shall be performed on the abstraction level of SFRs. If the developer defined security functionality groups (TSF-groups) in the TSS part of his Security Target, the evaluator should also consider them in order to get a better understanding for the context of the security services offered by the TOE.

to the elements of the Security Target having not already been covered by these PPs. However, in general the fact of compliance to a PP is not sufficient to avoid inconsistencies. Assume the following situation, where → stands for "complies with":

composite-ST → PP 1 → PP 2 ← base-ST

PP 1 may require any kind of conformance[9], but this does not affect the 'additional elements' that the base-ST may introduce beyond PP 2. In conclusion, these additions are not necessarily consistent with the composite-ST's additions chosen beyond PP 1: There is no scenario that ensures their consistency 'by construction'.

Note that consistency may be no direct matching: Objectives for the base component's environment may become objectives for the composite TOE.

12.10.2.3 Action ASE_COMP.1.1E

12.10.2.3.1 General

CC Part 3 ASE_COMP.1.1C: *The statement of compatibility shall describe the separation of the base component-TSF into relevant base component-TSF being used by the composite product Security Target and others.*

12.10.2.3.2 Work unit ASE_COMP.1-1

The evaluator ***shall check*** that the statement of compatibility describes the separation of the base component-TSF into *relevant* base component-TSF being used by the composite product Security Target and others.

Please note that TSF means 'TOE Security Functionality', whereby the TSF content is represented by SFRs. The respective TOE summary specification (TSS) shall provide, for each SFR, a description on how each SFR is met. The evaluator shall use this description in order to understand the contextual frame of the SFRs.

If the developer defined security functionality groups (TSF-groups) in the TSS part of his Security Target as such contextual frame of the SFRs, the evaluator should also consider them in order to get a better understanding for the context of the security services offered by the TOE.

This work unit relates to Step 2 of the Application Notes above. In order to determine the intersection area the evaluator considers the list of the base component-SFRs as given in the base component Security Target as single properties of the base component's security services.

These base component-SFRs shall be separated in three groups:

- **IP_SFR**: Irrelevant base component-SFRs not being used by the composite-ST;
- **RP_SFR-SERV**: *Relevant* base component-SFRs being used by the composite-ST to implement a security service with associated TSFI;
- **RP_SFR-MECH**: *Relevant* base component-SFRs being used by the composite-ST because of their security properties providing protection against attacks to the TOE as a whole and being addressed in ADV_ARC. These required security properties are a result of the security mechanisms and services that are implemented in the base component.

---

9 e.g. "strict" , "exact" or "demonstrable" according to the CC.

To give an example, assume that there are the following base component-SFRs: Cryptographic operations FCS_COP.1/RSA, FCS_COP.1/AES, FCS_COP.1/EC as well as tamper-resistance FPT_PHP.3 and limited capabilities and availability FMT_LIM.1 and FMT_LIM.2.

The second and third group RP_SFR-SERV and RP_SFR-MECH exactly represent the intersection area in question. For example, IP_SFR = {FCS_COP.1/AES}, RP_SFR-SERV = {FCS_COP.1/RSA, FCS_COP.1/EC} and RP_SFR-MECH = {FPT_PHP.3, FMT_LIM.1, FMT_LIM.2}, i.e. AES is not used by the composite TOE, but all other base component-SFRs are used. However, the RP_SFR-MECH cannot be directly connected to SFRs in the composite-ST.

The size of the overlapping area (i.e. the content of the group RP_SFR-SERV and RP_SFR-MECH) results from the concrete properties of the base-ST and the composite-ST. If the composite-ST does not use any property of the base-ST and, hence, the intersection area is an empty set (i.e. RP_SFR-MECH ∪ RP_SFR-SERV = {∅}), no further composite evaluation activities are necessary at all: In such a case there is a technical, but not a security composition of the base component and the dependent component.

The result of this work unit shall be integrated to the result of ASE_REQ.1.1E / ASE_REQ.1-16 (or the equivalent higher components if a higher assurance level is selected) and ASE_REQ.2.1E / ASE_REQ.2-13.

12.10.2.3.3 Work unit ASE_COMP.1-2

The evaluator ***shall examine*** the statement of compatibility to determine that the base component-TSF being used by the composite product Security Target are complete and consistent for the current composite product.

In order to determine the completeness of the list of the base component-TSF being used by the composite-ST, the evaluator shall verify that:

- {base component-SFRs} = IP_SFR ∪ RP_SFR-SERV ∪ RP_SFR-MECH;
- Elements that belong to RP_SFR-SERV and RP_SFR-MECH are taken into account during the evaluation of the composite TOE. The IP-SFR are obviously part of the base component but they are not considered during the evaluation of the composite TOE.

In order to determine the consistency of the list of the base component-TSF being used by the composite-ST, the evaluator shall verify that there are no ambiguities and contradictory statements.

The result of this work unit shall be integrated to the result of ASE_REQ.1.1E / ASE_REQ.1-16 (or the equivalent higher components if a higher assurance level is selected) and ASE_REQ.2.1E / ASE_REQ.2-13.

CC Part 3 ASE_COMP.1.2C: *The statement of compatibility between the composite product Security Target and the base component Security Target shall show (e.g. in form of a mapping) that the Security Targets of the composite product and of the related base component match, i.e. that there is no conflict between security environments, security objectives, and security requirements of the composite product Security Target and the base component Security Target. It may be provided by indicating the concerned elements directly in the composite product Security Target followed by explanatory text, if necessary.*

12.10.2.3.4 Work unit ASE_COMP.1-3

The evaluator ***shall check*** that the security assurance requirements of the composite evaluation represent a subset of the security assurance requirements of the base component and its evaluation.

This work unit relates to Step 2 of the Application Notes above. In order to ensure a sufficient degree of trustworthiness of the base component-TSF the evaluator compares the TOE security assurance requirements of the composite evaluation with those of the base component and its evaluation. The evaluator decides that the degree of trustworthiness of the base component-TSF is sufficient, if the composite product-SAR represent a subset of the base component-SAR:

base component-SAR $\supseteq$ composite product-SAR

E.g. the EAL chosen for the composite evaluation does not exceed the EAL applied to the evaluation of the base component.

The result of this work unit shall be integrated to the result of ASE_REQ.2.1E / ASE_REQ.2-12.

12.10.2.3.5 Work unit ASE_COMP.1-4

The evaluator ***shall examine*** the statement of compatibility to determine that all performed operations on the *relevant* TOE security functional requirements of the base component are appropriate for the composite product Security Target.

This work unit relates to Step 3 of the Application Notes above. The relevant TOE security functional requirements of the base component comprise at least the elements of the group RP_SFR-SERV (cf. the work unit ASE_COMP.1-1), but also the RP_SFR-MECH may be presented as relevant TOE security functional requirements. The non-relevant TOE security functional requirements belong to IP_SFR.

In order to perform this work unit the evaluator compares single parameters of the relevant SFRs of the base component with those of the composite evaluation. For example, the evaluator compares the properties of the respective components FCS_COP.1/RSA and determines that the composite-ST requires a key length of 2048 bit and the base-ST enforces the RSA-function with a key length of 1024 and 2048 bit, i.e. this parameter of the base component is appropriate for the composite-ST. Note, that the composite product-SFRs need not necessarily be the same as the base component-SFRs, e.g. a trusted channel (FTP_ITC.1) in the composite product can be built using an RSA implementation (FCS_COP.1/RSA) of the base component.

The result of this work unit shall be integrated to the result of ASE_REQ.2.1E / ASE_REQ.2-4.

12.10.2.3.6 Work unit ASE_COMP.1-5

The evaluator ***shall examine*** the statement of compatibility to determine that the *relevant* TOE security objectives of the base component are not contradictory to those of the composite product Security Target.

This work unit relates to Step 3 of the Application Notes above. The relevant TOE security objectives of the base-ST are those that are mapped to the relevant SFRs of the base-ST (cf. the work unit ASE_COMP.1-1).

In order to perform this work unit the evaluator compares the relevant TOE security objectives of the base-ST with those of the composite-ST and determines whether they are not contradictory.

The result of this work unit shall be integrated to the result of ASE_OBJ.2.1E / ASE_OBJ.2-1.

12.10.2.3.7 Work unit ASE_COMP.1-6

The evaluator ***shall examine*** the statement of compatibility to determine that the *significant* security objectives for the operational environment of the base component are not contradictory to those of the composite product Security Target.

This work unit relates to Step 3 of the Application Notes above. In order to determine which security objectives for the operational environment of the base-ST are significant for the composite-ST the evaluator analyses the objectives for the environment of the base-ST and their separation in the following groups:

- **IrOE:** The objectives for the environment being not relevant for the composite-ST, e.g. the objectives for the environment about the developing and manufacturing phases of the base component;

- **CfPOE:** The objectives for the environment being fulfilled by the composite-ST automatically. Such objectives for the environment of the base-ST can always be assigned to the TOE security objectives of the composite-ST. Due to this fact they will be fulfilled either by the composite product-SFRs or by the composite product-SARs automatically. To give an example, let there be an objective for the environment OE.Resp-Appl of the base-ST: 'All User Data are owned by Smartcard Embedded Software. Therefore, it must be assumed that security relevant User Data (especially cryptographic keys) are treated by the Smartcard Embedded Software as defined for the specific application context'. Furthermore, a TOE security objective OT.Key_Secrecy of the composite-ST is given: 'The secrecy of the signature private key used for signature generation is reasonably assured against attacks with a high attack potential.' If the private key is the only sensitive data element, then the objective for the environment OE.Resp-Appl is covered by the TOE security objective OT.Key_Secrecy automatically;

- **SgOE**: The remaining objectives for the environment of the base-ST belonging neither to the group IrOE nor CfOE Exactly this group makes up the significant objectives for the environment for the composite-ST, which shall be addressed in the composite-ST.

In order to accomplish this work unit the evaluator compares the significant security objectives for the operational environment of the base-ST with those of the composite-ST and determines whether they are not contradictory. If necessary, the significant security objectives for the operational environment of the base-ST shall be included into the composite-ST including the related assumptions from which the objectives for the environment are drawn. The inclusion is not necessary, if the composite-ST already contains equivalent (or similar) security objectives (covering all relevant aspects) and assumptions.

Since assurance of the development and manufacturing environment of the base component is confirmed by the base component's evaluation, the respective objectives for the base component, if any, belong to the group IrOE.

Assurance of development and manufacturing environment is usually completely addressed by the assurance class ALC, and, hence, requires no explicit security objective.

The result of this work unit shall be integrated to the result of ASE_OBJ.2.1E / ASE_OBJ.2-1 and ASE_OBJ.2.1E / ASE_OBJ.2-3.

13 Class ADV: Development

13.1 General

The purpose of the development activity is to assess the design documentation in terms of its adequacy to understand how the TSF meets the SFRs and how the implementation of these SFRs cannot be tampered with or bypassed. 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 interfaces of the TSF), a TOE design description (which describes the architecture of the TSF in terms of how it works in order to perform the functions related to the SFRs being claimed), and an implementation description (a source code level description). In addition, there is a security architecture description (which describes the architectural properties of the TSF to explain how its security enforcement cannot be compromised or bypassed), an internal description (which describes how the TSF was constructed in a manner that encourages understandability), and a security policy model (which formally describes the security policies enforced by the TSF).

13.2 Application notes

CC requirements for design documentation are levelled by the amount, and detail of information provided, and the degree of formality of the presentation of the information. At lower levels, the most security-critical portions of the TSF are described with the most detail, while less security-critical portions of the TSF are merely summarized; added assurance is gained by increasing the amount of information about the most security-critical portions of the TSF, and increasing the details about the less security-critical portions. The most assurance is achieved when thorough details and information of all portions are provided.

CC considers a document's degree of formality (that is, whether it is informal or semiformal) 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.

A functional specification provides a description of the purpose and method-of-use of 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 and how they are realised through interactions with the externally-visible interfaces to the TSF. If there is also audit functionality that detects and record the occurrences of such events, descriptions of this audit functionality would also be expected to be part of the functional specification; while this functionality is technically not directly invoked by the user at the external interface, it certainly is affected by what occurs at the user's external interface.

A design description is expressed in terms of logical divisions (subsystems or modules) that each provide a comprehensible service or function. For example, a firewall can be composed of subsystems that deal with packet filtering, with remote administration, with auditing, and with connection-level filtering. The 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.

13.3 Security Architecture (ADV_ARC)

13.3.1 Evaluation of sub-activity (ADV_ARC.1)

13.3.1.1 Objectives

The objective of this sub-activity is to determine whether the TSF is structured such that it cannot be tampered with or bypassed, and whether TSFs that provide security domains isolate those domains from each other.

13.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the implementation representation (if available);

f) the operational user guidance.

13.3.1.3 Application notes

The notions of self-protection, domain separation, and non-bypassability are distinct from security functionality expressed in CC Part 2 SFRs because self-protection and non-bypassability largely have no directly observable interface at the TSF. Rather, they are properties of the TSF that are achieved through the design of the TOE, and enforced by the correct implementation of that design. Also, the evaluation of these properties is less straight-forward than the evaluation of mechanisms; it is more difficult to check for the absence of functionality than for its presence. However, the determination that these properties are being satisfied is just as critical as the determination that the mechanisms are properly implemented.

The overall approach used is that the developer provides a TSF that meets the above-mentioned properties, and provides evidence (in the form of documentation) that can be analysed to show that the properties are indeed met. The evaluator has the responsibility for looking at the evidence and, coupled with other evidence delivered for the TOE, determining that the properties are achieved. The work units can be characterised as those detailing with what information has to be provided, and those dealing with the actual analysis the evaluator performs.

The security architecture description describes how domains are defined and how the TSF keeps them separate. It describes what prevents untrusted processes from getting to the TSF and modifying it. It describes what ensures that all resources under the TSF's control are adequately protected and that all actions related to the SFRs are mediated by the TSF. It explains any role the environment plays in any of these (e.g. presuming it gets correctly invoked by its underlying environment, how is its security functionality invoked?). In short, it explains how the TOE is considered to be providing any kind of *security* service.

The analyses the evaluator performs must be done in the context of all of the development evidence provided for the TOE, at the level of detail the evidence is provided. At lower assurance levels, there should not be the expectation that, for example, TSF self-protection is completely analysed, because only high-level design representations will be available. The evaluator also needs to be sure to use information gleaned from other portions of their analysis (e.g., analysis of

the TOE design) in making their assessments for the properties being examined in the following work units.

13.3.1.4 Action ADV_ARC.1.1E

13.3.1.4.1 General

CC Part 3 ADV_ARC.1.1C: *The security architecture description shall be at a level of detail commensurate with the description of the SFR-enforcing abstractions described in the TOE design document.*

13.3.1.4.2 Work unit ADV_ARC.1-1

The evaluator ***shall examine*** the security architecture description to determine that the information provided in the evidence is presented at a level of detail commensurate with the descriptions of the SFR-enforcing abstractions contained in the functional specification and TOE design document.

With respect to the functional specification, the evaluator should ensure that the self-protection functionality described cover those effects that are evident at the TSFI. Such a description can include protection placed upon the executable images of the TSF, and protection placed on objects (e.g., files used by the TSF). The evaluator ensures that the functionality that can be invoked through the TSFI is described.

If Evaluation of sub-activity (ADV_TDS.1) or Evaluation of sub-activity (ADV_TDS.2) is included, the evaluator ensures the security architecture description contains information on how any subsystems that contribute to TSF domain separation work.

If Evaluation of sub-activity (ADV_TDS.3) or higher is available, the evaluator ensures that the security architecture description also contains implementation-dependent information. For example, such a description can contain information pertaining to coding conventions for parameter checking that would prevent TSF compromises (e.g. buffer overflows), and information on stack management for call and return operations. The evaluator checks the descriptions of the mechanisms to ensure that the level of detail is such that there is little ambiguity between the description in the security architecture description and the implementation representation.

The evaluator action related to this work unit is assigned a fail verdict if the security architecture description mentions any module, subsystem, or interface that is not described in the functional specification or TOE design document.

CC Part 3 ADV_ARC.1.2C: *The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.*

13.3.1.4.3 Work unit ADV_ARC.1-2

The evaluator ***shall examine*** the security architecture description to determine that it describes the security domains maintained by the TSF.

Security domains refer to environments supplied by the TSF for use by potentially-harmful entities; for example, a typical secure operating system supplies a set of resources (address space, per-process environment variables) for use by processes with limited access rights and security properties. The evaluator determines that the developer's description of the security domains takes into account all of the SFRs claimed by the TOE.

For some TOEs such domains do not exist because all of the interactions available to users are severely constrained by the TSF. A packet-filter firewall is an example of such a TOE. Users on the

LAN or WAN do not interact with the TOE, so there need be no security domains; there are only data structures maintained by the TSF to keep the users' packets separated. The evaluator ensures that any claim that there are no domains is supported by the evidence and that no such domains are, in fact, available.

CC Part 3 ADV_ARC.1.3C: *The security architecture description shall describe how the TSF initialisation process is secure.*

13.3.1.4.4 Work unit ADV_ARC.1-3

The evaluator ***shall examine*** the security architecture description to determine that the initialisation process preserves security.

The information provided in the security architecture description relating to TSF initialisation is directed at the TOE components that are involved in bringing the TSF into an initial secure state (i.e. when all parts of the TSF are operational) when power-on or a reset is applied. This discussion in the security architecture description should list the system initialisation components and the processing that occurs in transitioning from the "down" state to the initial secure state.

It is often the case that the components that perform this initialisation function are not accessible after the secure state is achieved; if this is the case then the security architecture description identifies the components and explains how they are not reachable by untrusted entities after the TSF has been established. In this respect, the property that needs to be preserved is that these components either:

a) cannot be accessed by untrusted entities after the secure state is achieved, or;

b) if they provide interfaces to untrusted entities, these TSFI cannot be used to tamper with the TSF.

The TOE components related to TSF initialisation, then, are treated themselves as part of the TSF, and analysed from that perspective. It should be noted that even though these are treated as part of the TSF, it is likely that a justification (as allowed by TSF internals (ADV_INT)) can be made that they do not have to meet the internal structuring requirements of ADV_INT.

CC Part 3 ADV_ARC.1.4C: *The security architecture description shall demonstrate that the TSF protects itself from tampering.*

13.3.1.4.5 Work unit ADV_ARC.1-4

The evaluator ***shall examine*** the security architecture description to determine that it contains information sufficient to support a determination that the TSF is able to protect itself from tampering by untrusted active entities.

"Self-protection" refers to the ability of the TSF to protect itself from manipulation from external entities that may result in changes to the TSF. For TOEs that have dependencies on other IT entities, it is often the case that the TOE uses services supplied by the other IT entities in order to perform its functions. In such cases, the TSF alone does not protect itself because it depends on the other IT entities to provide some of the protection. For the purposes of the security architecture description, the notion of self-protection applies only to the services provided by the TSF through its TSFI, and not to services provided by underlying IT entities that it uses.

Self-protection is typically achieved by a variety of means, ranging from physical and logical restrictions on access to the TOE; to hardware-based means (e.g. "execution rings" and memory management functionality); to software-based means (e.g. boundary checking of inputs on a trusted server). The evaluator determines that all such mechanisms are described.

The evaluator determines that the design description covers how user input is handled by the TSF in such a way that the TSF does not subject itself to being corrupted by that user input. For example, the TSF can implement the notion of privilege and protect itself by using privileged-mode routines to handle user input. The TSF can make use of processor-based separation mechanisms such as privilege levels or rings. The TSF can implement software protection constructs or coding conventions that contribute to implementing separation of software domains, perhaps by delineating user address space from system address space. And the TSF might have reliance its environment to provide some support to the protection of the TSF.

All of the mechanisms contributing to the domain separation functions are described. The evaluator should use knowledge gained from other evidence (functional specification, TOE design, TSF internals description, other parts of the security architecture description, or implementation representation, as included in the assurance package for the TOE) in determining if any functionality contributing to self-protection was described that is not present in the security architecture description.

Accuracy of the description of the self-protection mechanisms is the property that the description faithfully describes what is implemented. The evaluator should use other evidence (functional specification, TOE design, TSF Internals documentation, other parts of the security architecture description, implementation representation, as included in the ST for the TOE) in determining whether there are discrepancies in any descriptions of the self-protection mechanisms. If Implementation representation (ADV_IMP) is included in the assurance package for the TOE, the evaluator will choose a sample of the implementation representation; the evaluator should also ensure that the descriptions are accurate for the sample chosen. If an evaluator cannot understand how a certain self-protection mechanism works or can work in the system architecture, it can be the case that the description is not accurate.

CC Part 3 ADV_ARC.1.5C: *The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.*

13.3.1.4.6 Work unit ADV_ARC.1-5

The evaluator ***shall examine*** the security architecture description to determine that it presents an analysis that adequately describes how the SFR-enforcing mechanisms cannot be bypassed.

Non-bypassability is a property that the security functionality of the TSF (as specified by the SFRs) is always invoked. For example, if access control to files is specified as a capability of the TSF via an SFR, there must be no interfaces through which files can be accessed without invoking the TSF's access control mechanism (such as an interface through which a raw disk access takes place).

Describing how the TSF mechanisms cannot be bypassed generally requires a systematic argument based on the TSF and the TSFIs. The description of how the TSF works (contained in the design decomposition evidence, such as the functional specification, TOE design documentation) - along with the information in the TSS - provides the background necessary for the evaluator to understand what resources are being protected and what security functions are being provided. The functional specification provides descriptions of the TSFIs through which the resources/functions are accessed.

The evaluator assesses the description provided (and other information provided by the developer, such as the functional specification) to ensure that no available interface can be used to bypass the TSF. This means that every available interface must be either unrelated to the SFRs that are claimed in the ST (and does not interact with anything that is used to satisfy SFRs) or else uses the security functionality that is described in other development evidence in the manner described. For example, a game would likely be unrelated to the SFRs, so there must be an explanation of how it cannot affect security. Access to user data, however, is likely to be related

to access control SFRs, so the explanation would describe how the security functionality works when invoked through the data-access interfaces. Such a description is needed for every available interface.

An example of a description follows. Suppose the TSF provides file protection. Further suppose that although the "traditional" system call TSFIs for open, read, and write invoke the file protection mechanism described in the TOE design, there exists a TSFI that allows access to a batch job facility (creating batch jobs, deleting jobs, modifying unprocessed jobs). The evaluator should be able to determine from the vendor-provided description that this TSFI invokes the same protection mechanisms as do the "traditional" interfaces. This can be done, for example, by referencing the appropriate subclauses of the TOE design that discuss *how* the batch job facility TSFI achieves its security objectives.

Using this same example, suppose there is a TSFI whose sole purpose is to display the time of day. The evaluator should determine that the description adequately argues that this TSFI is not capable of manipulating any protected resources and should not invoke any security functionality.

Another example of bypass is when the TSF is supposed to maintain confidentiality of a cryptographic key (one is allowed to use it for cryptographic operations, but is not allowed to read/write it). If an attacker has direct physical access to the device, they might be able to examine side-channels such as the power usage of the device, the exact timing of the device, or even any electromagnetic emanations of the device and, from this, infer the key.

If such side-channels may be present, the demonstration should address the mechanisms that prevent these side-channels from occurring, such as random internal clocks, dual-line technology. Verification of these mechanisms would be verified by a combination of purely design-based arguments and testing.

For a final example using security functionality rather than a protected resource, consider an ST that contains FCO_NRO.2 Enforced proof of origin, which requires that the TSF provides evidence of origination for information types specified in the ST. Suppose that the "information types" included all information that is sent by the TOE via e-mail. In this case, the evaluator should examine the description to ensure that all TSFI that can be invoked to send e-mail perform the "evidence of origination generation" function are detailed. The description can point to user guidance to show all places where e-mail can originate (e.g., e-mail program, notification from scripts/batch jobs) and then how each of these places invokes the evidence generation function.

The evaluator should also ensure that the description is comprehensive, in that each interface is analysed with respect to the entire set of claimed SFRs. This may require the evaluator to examine supporting information (functional specification, TOE design, other parts of the security architecture description, operational user guidance, and perhaps even the implementation representation, as provided for the TOE) to determine that the description has correctly capture all aspects of an interface. The evaluator should consider what SFRs each TSFI can affect (from the description of the TSFI and its implementation in the supporting documentation), and then examine the description to determine whether it covers those aspects.

13.4 Functional specification (ADV_FSP)

13.4.1 Evaluation of sub-activity (ADV_FSP.1)

13.4.1.1 Objectives

The objective of this sub-activity is to determine whether the developer has provided a high-level description of at least the SFR-enforcing and SFR-supporting TSFIs, in terms of descriptions of their parameters. There is no other required evidence that can be expected to be available to

measure the accuracy of these descriptions; the evaluator merely ensures the descriptions seem plausible.

13.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the operational user guidance.

13.4.1.3 Action ADV_FSP.1.1E

13.4.1.3.1 General

CC Part 3 ADV_FSP.1.1C: *The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.*

13.4.1.3.2 Work unit ADV_FSP.1-1

The evaluator ***shall examine*** the functional specification to determine that it states the purpose of each SFR-supporting and SFR-enforcing TSFI.

The purpose of a TSFI is a general statement summarising the functionality provided by the interface. It is not intended to be a complete statement of the actions and results related to the interface, but rather a statement to help the reader understand in general what the interface is intended to be used for. The evaluator should not only determine that the purpose exists, but also that it accurately reflects the TSFI by taking into account other information about the interface, such as the description of the parameters; this can be done in association with other work units for this component.

If an action available through an interface plays a role in enforcing any security policy on the TOE (that is, if one of the actions of the interface can be traced to one of the SFRs levied on the TSF), then that interface is *SFR-enforcing*. Such policies are not limited to the access control policies, but also refer to any functionality specified by one of the SFRs contained in the ST. Note that it is possible that an interface may have various actions and results, some of which may be SFR-enforcing and some of which may not.

Interfaces to (or actions available through an interface relating to) actions that SFR-enforcing functionality depends on, but need only to function correctly in order for the security policies of the TOE to be preserved, are termed *SFR supporting*. Interfaces to actions on which SFR-enforcing functionality has no dependence are termed *SFR non-interfering*.

It should be noted that in order for an interface to be SFR supporting or SFR non-interfering it must have *no* SFR-enforcing actions or results. In contrast, an SFR-enforcing interface may have SFR-supporting actions (for example, the ability to set the system clock may be an SFR-enforcing action of an interface, but if that same interface is used to display the system date that action may only be SFR supporting). An example of a purely SFR-supporting interface is a system call interface that is used both by untrusted users and by a portion of the TSF that is running in user mode.

At this level, it is unlikely that a developer will have expended effort to label interfaces as SFR-enforcing and SFR-supporting. In the case that this has been done, the evaluator should verify to the extent that supporting documentation (e.g., operational user guidance) allows that this identification is correct. Note that this identification activity is necessary for several work units for this component.

In the more likely case that the developer has not labelled the interfaces, the evaluator must perform their own identification of the interfaces first, and then determine whether the required information (for this work unit, the purpose) is present. Again, because of the lack of supporting evidence this identification will be difficult and have low assurance that all appropriate interfaces have been correctly identified, but nonetheless the evaluator examines other evidence available for the TOE to ensure as complete coverage as is possible.

13.4.1.3.3 Work unit ADV_FSP.1-2

The evaluator ***shall examine*** the functional specification to determine that the method of use for each SFR-supporting and SFR-enforcing TSFI is given.

See work unit ADV_FSP.1-1 for a discussion on the identification of SFR-supporting and SFR-enforcing TSFI.

The method of use for a TSFI summarizes how the interface is manipulated in order to invoke the actions and obtain the results associated with the TSFI. The evaluator should be able to determine, from reading this material in the functional specification, how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each TSFI, as it may be possible to describe in general how kernel calls are invoked, for instance, and then identify each interface using that general style. Different types of interfaces will require different method of use specifications. APIs, network protocol interfaces, system configuration parameters, and hardware bus interfaces all have very different methods of use, and this should be taken into account by the developer when developing the functional specification, as well as by the evaluator evaluating the functional specification.

For administrative interfaces, whose functionality is documented as being inaccessible to untrusted users, the evaluator ensures that the method of making the functions inaccessible is described in the functional specification. It should be noted that this inaccessibility needs to be tested by the developer in their test suite.

CC Part 3 ADV_FSP.1.2C: *The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.*

13.4.1.3.4 Work unit ADV_FSP.1-3

The evaluator ***shall examine*** the presentation of the TSFI to determine that it identifies all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

See work unit ADV_FSP.1-1 for a discussion on the identification of SFR-supporting and SFR-enforcing TSFI.

The evaluator examines the functional specification to ensure that all of the parameters are described for identified TSFI. Parameters are explicit inputs or outputs to an interface that control the behaviour of that interface. For examples, parameters are the arguments supplied to an API; the various fields in packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

While difficult to obtain much assurance that all parameters for the applicable TSFI have been identified, the evaluator should also check other evidence provided for the evaluation (e.g., operational user guidance) to see if behaviour or additional parameters are described there but not in the functional specification.

CC Part 3 ADV_FSP.1.3C: *The functional specification shall provide rationale for the implicit categorisation of interfaces as SFR-non-interfering.*

13.4.1.3.5 Work unit ADV_FSP.1-4

The evaluator ***shall examine*** the rationale provided by the developer for the implicit categorisation of interfaces as SFR-non-interfering to determine that it is accurate.

In the case where the developer has provided adequate documentation to perform the analysis called for by the rest of the work units for this component without explicitly identifying SFR-enforcing and SFR-supporting interfaces, this work unit should be considered satisfied.

This work unit is intended to apply to cases where the developer has not described a portion of the TSFI, claiming that it is SFR-non-interfering and therefore not subject to other requirements of this component. In such a case, the developer provides a rationale for this characterisation in sufficient detail such that the evaluator understands the rationale, the characteristics of the interfaces affected (e.g., their high-level function with respect to the TOE, such as "colour palette manipulation"), and that the claim that these are SFR-non-interfering is supported. Given the level of assurance the evaluator should not expect more detail than is provided for the SFR-enforcing or SFR-supporting interfaces, and in fact the detail should be much less. In most cases, individual interfaces should not need to be addressed in the developer-provided rationale subclause.

CC Part 3 ADV_FSP.1.4C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.1.3.6 Work unit ADV_FSP.1-5

The evaluator ***shall check*** that the tracing links the SFRs to the corresponding TSFIs.

The tracing is provided by the developer to serve as a guide to which SFRs are related to which TSFIs. This tracing can be as simple as a table; it is used as input to the evaluator for use in the following work units, in which the evaluator verifies its completeness and accuracy.

13.4.1.4 Action ADV_FSP.1.2E

13.4.1.4.1 Work unit ADV_FSP.1-6

The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the SFRs.

To ensure that all SFRs are covered by the functional specification, as well as the test coverage analysis, the evaluator may build upon the developer's tracing (see ADV_FSP.1-5 a map between the TOE security functional requirements and the TSFI). Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 assignment, and these ten rules were covered by three different TSFI, it would be inadequate for the evaluator to map FDP_ACC.1 to TSFI A, B, and C and claim they had completed the work unit. Instead, the evaluator would map FDP_ACC.1 (rule 1) to TSFI A; FDP_ACC.1 (rule 2) to TSFI B; etc. It can also be the case that the interface is a wrapper interface (e.g., IOCTL), in which case the mapping would need to be specific to certain set of parameters for a given interface.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that they completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST. It is also important to note that since the parameters

associated with TSFIs must be fully specified, the evaluator should be able to determine if all aspects of an SFR appear to be implemented at the interface level.

13.4.1.4.2 Work unit ADV_FSP.1-7

The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the SFRs.

For each functional requirement in the ST that results in effects visible at the TSF boundary, the information in the associated TSFI for that requirement specifies the required functionality described by the requirement. For example, if the ST contains a requirement for access control lists, and the only TSFI that map to that requirement specify functionality for Unix-style protection bits, then the functional specification is not accurate with respect to the requirements.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that the evaluator completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST.

13.4.2 Evaluation of sub-activity (ADV_FSP.2)

13.4.2.1 Objectives

The objective of this sub-activity is to determine whether the developer has provided a description of the TSFIs in terms of their purpose, method of use, and parameters. In addition, the SFR-enforcing actions, results and error messages of each TSFI that is SFR-enforcing are also described.

13.4.2.2 Input

The evaluation evidence for this sub-activity that is required by the work-units is:

a) the ST;

b) the functional specification;

c) the TOE design.

The evaluation evidence for this sub-activity that is used if included in the ST for the TOE is:

a) the security architecture description;

b) the operational user guidance.

13.4.2.3 Action ADV_FSP.2.1E

13.4.2.3.1 General

CC Part 3 ADV_FSP.2.1C: *The functional specification shall completely represent the TSF.*

13.4.2.3.2 Work unit ADV_FSP.2-1

The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

The identification of the TSFI is a necessary prerequisite to all other activities in this sub-activity. The TSF must be identified (done as part of the TOE design (ADV_TDS) work units) in order to identify the TSFI. This activity can be done at a high level to ensure that no large groups of

interfaces have been missed (network protocols, hardware interfaces, configuration files), or at a low level as the evaluation of the functional specification proceeds.

In making an assessment for this work unit, the evaluator determines that all portions of the TSF are addressed in terms of the interfaces listed in the functional specification. All portions of the TSF should have a corresponding interface description, or if there are no corresponding interfaces for a portion of the TSF, the evaluator determines that that is acceptable.

CC Part 3 ADV_FSP.2.2C: *The functional specification shall describe the purpose and method of use for all TSFI.*

13.4.2.3.3 Work unit ADV_FSP.2-2

The evaluator ***shall examine*** the functional specification to determine that it states the purpose of each TSFI.

The purpose of a TSFI is a general statement summarising the functionality provided by the interface. It is not intended to be a complete statement of the actions and results related to the interface, but rather a statement to help the reader understand in general what the interface is intended to be used for. The evaluator should not only determine that the purpose exists, but also that it accurately reflects the TSFI by taking into account other information about the interface, such as the description of actions and error messages.

13.4.2.3.4 Work unit ADV_FSP.2-3

The evaluator ***shall examine*** the functional specification to determine that the method of use for each TSFI is given.

The method of use for a TSFI summarizes how the interface is manipulated in order to invoke the actions and obtain the results associated with the TSFI. The evaluator should be able to determine, from reading this material in the functional specification, how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each TSFI, as it may be possible to describe in general how kernel calls are invoked, for instance, and then identify each interface using that general style. Different types of interfaces will require different method of use specifications. APIs, network protocol interfaces, system configuration parameters, and hardware bus interfaces all have very different methods of use, and this should be taken into account by the developer when developing the functional specification, as well as by the evaluator evaluating the functional specification.

For administrative interfaces, whose functionality is documented as being inaccessible to untrusted users, the evaluator ensures that the method of making the functions inaccessible is described in the functional specification. It should be noted that this inaccessibility needs to be tested by the developer in their test suite.

The evaluator should not only determine that the set of method of use descriptions exist, but also that they accurately cover each TSFI.

CC Part 3 ADV_FSP.2.3C: *The functional specification shall identify and describe all parameters associated with each TSFI.*

13.4.2.3.5 Work unit ADV_FSP.2-4

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely identifies all parameters associated with every TSFI.

The evaluator examines the functional specification to ensure that all of the parameters are described for each TSFI. Parameters are explicit inputs or outputs to an interface that control the

behaviour of that interface. For examples, parameters are the arguments supplied to an API; the various fields in packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

In order to determine that all of the parameters are present in the TSFI, the evaluator should examine the rest of the interface description (actions, error messages, etc.) to determine if the effects of the parameter are accounted for in the description. The evaluator should also check other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

13.4.2.3.6 Work unit ADV_FSP.2-5

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all parameters associated with every TSFI.

Once all of the parameters have been identified, the evaluator needs to ensure that they are accurately described, and that the description of the parameters is complete. A parameter description tells what the parameter is in some meaningful way. For instance, the interface *foo(i)* can be described as having "parameter i which is an integer"; this is not an acceptable parameter description. A description such as "parameter i is an integer that indicates the number of users currently logged in to the system" is much more acceptable.

In order to determine that the description of the parameters is complete, the evaluator should examine the rest of the interface description (purpose, method of use, actions, error messages, etc.) to determine if the descriptions of the parameter(s) are accounted for in the description. The evaluator should also check other evidence provided (e.g., TOE design, architectural design, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

CC Part 3 ADV_FSP.2.4C: *For each SFR-enforcing TSFI, the functional specification shall describe the SFR-enforcing actions associated with the TSFI.*

13.4.2.3.7 Work unit ADV_FSP.2-6

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes the SFR-enforcing actions associated with the SFR-enforcing TSFIs.

If an action available through an interface can be traced to one of the SFRs levied on the TSF, then that interface is *SFR-enforcing*. Such policies are not limited to the access control policies, but also refer to any functionality specified by one of the SFRs contained in the ST. Note that it is possible that an interface may have various actions and results, some of which may be SFR-enforcing and some of which may not.

The developer is not required to "label" interfaces as SFR-enforcing, and likewise is not required to identify actions available through an interface as SFR-enforcing. It is the evaluator's responsibility to examine the evidence provided by the developer and determine that the required information is present. In the case where the developer has identified the SFR-enforcing TSFI and SFR-enforcing actions available through those TSFI, the evaluator must judge completeness and accuracy based on other information supplied for the evaluation (e.g., TOE design, security architecture description, operational user guidance), and on the other information presented for the interfaces (parameters and parameter descriptions, error messages, etc.).

In this case (where the developer has provided only the SFR-enforcing information for SFR-enforcing TSFI) the evaluator also ensures that no interfaces have been mis-categorised. This is done by examining other information supplied for the evaluation (e.g., TOE design, security

architecture description, operational user guidance), and the other information presented for the interfaces (parameters and parameter descriptions, for example) not labelled as SFR-enforcing.

In the case where the developer has provided the same level of information on all interfaces, the evaluator performs the same type of analysis mentioned in the previous paragraphs. The evaluator should determine which interfaces are SFR-enforcing and which are not, and subsequently ensure that the SFR-enforcing aspects of the SFR-enforcing actions are appropriately described.

The SFR-enforcing actions are those that are visible at any external interface and that provide for the enforcement of the SFRs being claimed. For example, if audit requirements are included in the ST, then audit-related actions would be SFR-enforcing and therefore must be described, even if the result of that action is generally not visible through the invoked interface (as is often the case with audit, where a user action at one interface would produce an audit record visible at another interface).

The level of description that is required is that sufficient for the reader to understand what role the TSFI actions play with respect to the SFR. The evaluator should keep in mind that the description should be detailed enough to support the generation (and assessment) of test cases against that interface. If the description is unclear or lacking detail such that meaningful testing cannot be conducted against the TSFI, it is likely that the description is inadequate.

CC Part 3 ADV_FSP.2.5C: *For each SFR-enforcing TSFI, the functional specification shall describe direct error messages resulting from processing associated with the SFR-enforcing actions.*

13.4.2.3.8 Work unit ADV_FSP.2-7

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes error messages that may result from SFR-enforcing actions associated with each SFR-enforcing TSFI.

This work unit should be performed in conjunction with, or after, work unit ADV_FSP.2-6 in order to ensure the set of SFR-enforcing TSFI and SFR-enforcing actions is correctly identified. The developer may provide more information than is required (for example, all error messages associated with each interface), in which the case the evaluator should restrict their assessment of completeness and accuracy to only those that they determine to be associated with SFR-enforcing actions of SFR-enforcing TSFI.

Errors can take many forms, depending on the interface being described. For an API, the interface itself may return an error code, set a global error condition, or set a certain parameter with an error code. For a configuration file, an incorrectly configured parameter may cause an error message to be written to a log file. For a hardware PCI card, an error condition may raise a signal on the bus, or trigger an exception condition to the CPU.

Errors (and the associated error messages) come about through the invocation of an interface. The processing that occurs in response to the interface invocation may encounter error conditions, which trigger (through an implementation-specific mechanism) an error message to be generated. In some instances, this may be a return value from the interface itself; in other instances a global value may be set and checked after the invocation of an interface. It is likely that a TOE will have a number of low-level error messages that may result from fundamental resource conditions, such as "disk full" or "resource locked". While these error messages may map to a large number of TSFI, they can be used to detect instances where detail from an interface description has been omitted. For instance, a TSFI that produces a "disk full" message, but has no obvious description of why that TSFI should cause an access to the disk in its description of actions, can cause the evaluator to examine other evidence (Security Architecture (ADV_ARC), TOE design (ADV_TDS)) related that TSFI to determine if the description is accurate.

In order to determine that the description of the error messages of a TSFI is accurate and complete, the evaluator measures the interface description against the other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance), as well as other evidence available for that TSFI (parameters, analysis from work unit ADV_FSP.2-6).

CC Part 3 ADV_FSP.2.6C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.2.3.9 Work unit ADV_FSP.2-8

The evaluator ***shall check*** that the tracing links the SFRs to the corresponding TSFIs.

The tracing is provided by the developer to serve as a guide to which SFRs are related to which TSFIs. This tracing can be as simple as a table; it is used as input to the evaluator for use in the following work units, in which the evaluator verifies its completeness and accuracy.

13.4.2.4 Action ADV_FSP.2.2E

13.4.2.4.1 Work unit ADV_FSP.2-9

The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the SFRs.

To ensure that all SFRs are covered by the functional specification, as well as the test coverage analysis, the evaluator may build upon the developer's tracing (see ADV_FSP.2-8 a map between the TOE security functional requirements and the TSFI. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 assignment, and these ten rules were covered by three different TSFI, it would be inadequate for the evaluator to map FDP_ACC.1 to TSFI A, B, and C and claim they had completed the work unit. Instead, the evaluator would map FDP_ACC.1 (rule 1) to TSFI A; FDP_ACC.1 (rule 2) to TSFI B; etc. It can also be the case that the interface is a wrapper interface (e.g., IOCTL), in which case the mapping would need to be specific to certain set of parameters for a given interface.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that they completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST. It is also important to note that since the parameters, actions, and error messages associated with TSFIs must be fully specified, the evaluator should be able to determine if all aspects of an SFR appear to be implemented at the interface level.

13.4.2.4.2 Work unit ADV_FSP.2-10

The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the SFRs.

For each functional requirement in the ST that results in effects visible at the TSF boundary, the information in the associated TSFI for that requirement specifies the required functionality described by the requirement. For example, if the ST contains a requirement for access control lists, and the only TSFI that map to that requirement specify functionality for Unix-style protection bits, then the functional specification is not accurate with respect to the requirements.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that the evaluator completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST.

13.4.3 Evaluation of sub-activity (ADV_FSP.3)

13.4.3.1 Objectives

The objective of this sub-activity is to determine whether the developer has provided a description of the TSFIs in terms of their purpose, method of use, and parameters. In addition, the actions, results and error messages of each TSFI are also described sufficiently that it can be determined whether they are SFR-enforcing, with the SFR-enforcing TSFI being described in more detail than other TSFIs.

13.4.3.2 Input

The evaluation evidence for this sub-activity that is required by the work-units is:

a) the ST;

b) the functional specification;

c) the TOE design.

The evaluation evidence for this sub-activity that is used if included in the ST for the TOE is:

a) the security architecture description;

b) the implementation representation;

c) the TSF internals description;

d) the operational user guidance.

13.4.3.3 Action ADV_FSP.3.1E

13.4.3.3.1 General

CC Part 3 ADV_FSP.3.1C: *The functional specification shall completely represent the TSF.*

13.4.3.3.2 Work unit ADV_FSP.3-1

The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

The identification of the TSFI is a necessary prerequisite to all other activities in this sub-activity. The TSF must be identified (done as part of the TOE design (ADV_TDS) work units) in order to identify the TSFI. This activity can be done at a high level to ensure that no large groups of interfaces have been missed (network protocols, hardware interfaces, configuration files), or at a low level as the evaluation of the functional specification proceeds.

In making an assessment for this work unit, the evaluator determines that all portions of the TSF are addressed in terms of the interfaces listed in the functional specification. All portions of the TSF should have a corresponding interface description, or if there are no corresponding interfaces for a portion of the TSF, the evaluator determines that that is acceptable.

CC Part 3 ADV_FSP.3.2C: *The functional specification shall describe the purpose and method of use for all TSFI.*

13.4.3.3.3 Work unit ADV_FSP.3-2

The evaluator ***shall examine*** the functional specification to determine that it states the purpose of each TSFI.

The purpose of a TSFI is a general statement summarising the functionality provided by the interface. It is not intended to be a complete statement of the actions and results related to the interface, but rather a statement to help the reader understand in general what the interface is intended to be used for. The evaluator should not only determine that the purpose exists, but also that it accurately reflects the TSFI by taking into account other information about the interface, such as the description of actions and error messages.

13.4.3.3.4 Work unit ADV_FSP.3-3

The evaluator ***shall examine*** the functional specification to determine that the method of use for each TSFI is given.

The method of use for a TSFI summarizes how the interface is manipulated in order to invoke the actions and obtain the results associated with the TSFI. The evaluator should be able to determine, from reading this material in the functional specification, how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each TSFI, as it may be possible to describe in general how kernel calls are invoked, for instance, and then identify each interface using that general style. Different types of interfaces will require different method of use specifications. APIs, network protocol interfaces, system configuration parameters, and hardware bus interfaces all have very different methods of use, and this should be taken into account by the developer when developing the functional specification, as well as by the evaluator evaluating the functional specification.

For administrative interfaces whose functionality is documented as being inaccessible to untrusted users, the evaluator ensures that the method of making the functions inaccessible is described in the functional specification. It should be noted that this inaccessibility needs to be tested by the developer in their test suite.

The evaluator should not only determine that the set of method of use descriptions exist, but also that they accurately cover each TSFI.

CC Part 3 ADV_FSP.3.3C: *The functional specification shall identify and describe all parameters associated with each TSFI.*

13.4.3.3.5 Work unit ADV_FSP.3-4

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely identifies all parameters associated with every TSFI.

The evaluator examines the functional specification to ensure that all of the parameters are described for each TSFI. Parameters are explicit inputs or outputs to an interface that control the behaviour of that interface. For examples, parameters are the arguments supplied to an API; the various fields in packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

In order to determine that all of the parameters are present in the TSFI, the evaluator should examine the rest of the interface description (actions, error messages, etc.) to determine if the effects of the parameter are accounted for in the description. The evaluator should also check other evidence provided for the evaluation (e.g., TOE design, security architecture description,

operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

13.4.3.3.6 Work unit ADV_FSP.3-5

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all parameters associated with every TSFI.

Once all of the parameters have been identified, the evaluator needs to ensure that they are accurately described, and that the description of the parameters is complete. A parameter description tells what the parameter is in some meaningful way. For instance, the interface *foo(i)* can be described as having "parameter i which is an integer"; this is not an acceptable parameter description. A description such as "parameter i is an integer that indicates the number of users currently logged in to the system" is much more acceptable.

In order to determine that the description of the parameters is complete, the evaluator should examine the rest of the interface description (purpose, method of use, actions, error messages, etc.) to determine if the descriptions of the parameter(s) are accounted for in the description. The evaluator should also check other evidence provided (e.g., TOE design, architectural design, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

CC Part 3 ADV_FSP.3.4C: *For each SFR-enforcing TSFI, the functional specification shall describe the SFR-enforcing actions associated with the TSFI.*

13.4.3.3.7 Work unit ADV_FSP.3-6

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes the SFR-enforcing actions associated with the SFR-enforcing TSFIs.

If an action available through an interface plays a role in enforcing any security policy on the TOE (that is, if one of the actions of the interface can be traced to one of the SFRs levied on the TSF), then that interface is *SFR-enforcing*. Such policies are not limited to the access control policies, but also refer to any functionality specified by one of the SFRs contained in the ST. Note that it is possible that an interface may have various actions and results, some of which may be SFR-enforcing and some of which may not.

The developer is not required to "label" interfaces as SFR-enforcing, and likewise is not required to identify actions available through an interface as SFR-enforcing. It is the evaluator's responsibility to examine the evidence provided by the developer and determine that the required information is present. In the case where the developer has identified the SFR-enforcing TSFI and SFR-enforcing actions available through those TSFI, the evaluator must judge completeness and accuracy based on other information supplied for the evaluation (e.g., TOE design, security architecture description, operational user guidance), and on the other information presented for the interfaces (parameters and parameter descriptions, error messages, etc.).

In this case (developer has provided only the SFR-enforcing information for SFR-enforcing TSFI) the evaluator also ensures that no interfaces have been mis-categorised. This is done by examining other information supplied for the evaluation (e.g., TOE design, security architecture description, operational user guidance), and the other information presented for the interfaces (parameters and parameter descriptions, for example) not labelled as SFR-enforcing. The analysis done for work units ADV_FSP.3-7 and ADV_FSP.3-8 are also used in making this determination.

In the case where the developer has provided the same level of information on all interfaces, the evaluator performs the same type of analysis mentioned in the previous paragraphs. The

evaluator should determine which interfaces are SFR-enforcing and which are not, and subsequently ensure that the SFR-enforcing aspects of the SFR-enforcing actions are appropriately described. Note that in this case, the evaluator should be able to perform the bulk of the work associated with work unit ADV_FSP.3-8 in the course of performing this SFR-enforcing analysis.

The SFR-enforcing actions are those that are visible at any external interface and that provide for the enforcement of the SFRs being claimed. For example, if audit requirements are included in the ST, then audit-related actions would be SFR-enforcing and therefore must be described, even if the result of that action is generally not visible through the invoked interface (as is often the case with audit, where a user action at one interface would produce an audit record visible at another interface).

The level of description that is required is that sufficient for the reader to understand what role the TSFI actions play with respect to the SFR. The evaluator should keep in mind that the description should be detailed enough to support the generation (and assessment) of test cases against that interface. If the description is unclear or lacking detail such that meaningful testing cannot be conducted against the TSFI, it is likely that the description is inadequate.

CC Part 3 ADV_FSP.3.5C: *For each SFR-enforcing TSFI, the functional specification shall describe direct error messages resulting from SFR-enforcing actions and exceptions associated with invocation of the TSFI.*

13.4.3.3.8 Work unit ADV_FSP.3-7

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes error messages that may result from an invocation of each SFR-enforcing TSFI.

This work unit should be performed in conjunction with, or after, work unit ADV_FSP.3-6 in order to ensure the set of SFR-enforcing TSFI is correctly identified. The evaluator should note that the requirement and associated work unit is that all direct error messages associated with an SFR-enforcing TSFI must be described, that are associated with SFR-enforcing actions. This is because at this level of assurance, the "extra" information provided by the error message descriptions should be used in determining whether all of the SFR-enforcing aspects of an interface have been appropriately described. For instance, if an error message associated with a TSFI (e.g., "access denied") indicated that an SFR-enforcing decision or action had taken place, but in the description of the SFR-enforcing actions there was no mention of that particular SFR-enforcing mechanism, then the description may not be complete.

Errors can take many forms, depending on the interface being described. For an API, the interface itself may return an error code, set a global error condition, or set a certain parameter with an error code. For a configuration file, an incorrectly configured parameter may cause an error message to be written to a log file. For a hardware PCI card, an error condition may raise a signal on the bus, or trigger an exception condition to the CPU.

Errors (and the associated error messages) come about through the invocation of an interface. The processing that occurs in response to the interface invocation may encounter error conditions, which trigger (through an implementation-specific mechanism) an error message to be generated. In some instances this may be a return value from the interface itself; in other instances a global value may be set and checked after the invocation of an interface. It is likely that a TOE will have a number of low-level error messages that may result from fundamental resource conditions, such as "disk full" or "resource locked". While these error messages may map to a large number of TSFI, they can be used to detect instances where detail from an interface description has been omitted. For instance, a TSFI that produces a "disk full" message, but has no obvious description of why that TSFI should cause an access to the disk in its description of

actions, can cause the evaluator to examine other evidence (Security Architecture (ADV_ARC), TOE design (ADV_TDS)) related that TSFI to determine if the description is accurate.

In order to determine that the description of the error messages of a TSFI is accurate and complete, the evaluator measures the interface description against the other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance), as well as for other evidence supplied for that TSFI (description of SFR-enforcing actions, summary of SFR-supporting and SFR-non-interfering actions and results).

CC Part 3 ADV_FSP.3.6C: *The functional specification shall summarize the SFR-supporting and SFR-non-interfering actions associated with each TSFI.*

13.4.3.3.9 Work unit ADV_FSP.3-8

The evaluator ***shall examine*** the presentation of the TSFI to determine that it summarizes the SFR-supporting and SFR-non-interfering actions associated with each TSFI.

The purpose of this work unit is to supplement the details about the SFR-enforcing actions (provided in work unit ADV_FSP.3-6) with a summary of the remaining actions (i.e., those that are not SFR-enforcing). This covers *all* SFR-supporting and SFR-non-interfering actions, whether invokable through SFR-enforcing TSFI or through SFR-supporting or SFR-non-interfering TSFI. Such a summary about all SFR-supporting and SFR-non-interfering actions helps to provide a more complete picture of the functions provided by the TSF, and is to be used by the evaluator in determining whether an action or TSFI may have been mis-categorised.

The information to be provided is more abstract than that required for SFR-enforcing actions. While it should still be detailed enough so that the reader can understand what the action does, the description does not have to be detailed enough to support writing tests against it, for instance. For the evaluator, the key is that the information must be sufficient to make a positive determination that the action is SFR-supporting or SFR-non-interfering. If that level of information is missing, the summary is insufficient and more information must be obtained.

CC Part 3 ADV_FSP.3.7C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.3.3.10 Work unit ADV_FSP.3-9

The evaluator ***shall check*** that the tracing links the SFRs to the corresponding TSFIs.

The tracing is provided by the developer to serve as a guide to which SFRs are related to which TSFIs. This tracing can be as simple as a table; it is used as input to the evaluator for use in the following work units, in which the evaluator verifies its completeness and accuracy.

13.4.3.4 Action ADV_FSP.3.2E

13.4.3.4.1 Work unit ADV_FSP.3-10

The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the SFRs.

To ensure that all SFRs are covered by the functional specification, as well as the test coverage analysis, the evaluator may build upon the developer's tracing (see ADV_FSP.3-9 a map between the TOE security functional requirements and the TSFI. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 assignment, and these ten rules were covered by three different TSFI, it would be inadequate for the evaluator to map FDP_ACC.1 to TSFI A, B, and C and claim they had completed the work unit. Instead, the evaluator would map FDP_ACC.1 (rule 1) to TSFI A; FDP_ACC.1 (rule 2) to TSFI B; etc. It can also be the case that the interface is a wrapper interface (e.g., IOCTL), in which case the mapping would need to be specific to certain set of parameters for a given interface.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that they completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST. It is also important to note that since the parameters, actions, and error messages associated with TSFIs must be fully specified, the evaluator should be able to determine if all aspects of an SFR appear to be implemented at the interface level.

13.4.3.4.2 Work unit ADV_FSP.3-11

The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the SFRs.

For each functional requirement in the ST that results in effects visible at the TSF boundary, the information in the associated TSFI for that requirement specifies the required functionality described by the requirement. For example, if the ST contains a requirement for access control lists, and the only TSFI that map to that requirement specify functionality for Unix-style protection bits, then the functional specification is not accurate with respect to the requirements.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that the evaluator completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST.

13.4.4 Evaluation of sub-activity (ADV_FSP.4)

13.4.4.1 Objectives

The objective of this sub-activity is to determine whether the developer has completely described all of the TSFI in a manner such that the evaluator is able to determine whether the TSFI are completely and accurately described, and appears to implement the security functional requirements of the ST.

13.4.4.2 Input

The evaluation evidence for this sub-activity that is required by the work-units is:

a) the ST;

b) the functional specification;

c) the TOE design.

The evaluation evidence for this sub-activity that is used if included in the ST for the TOE is:

a) the security architecture description;

b) the implementation representation;

c) the TSF internals description;

d) the operational user guidance.

13.4.4.3 Application notes

The functional specification describes the interfaces to the TSF (the TSFI) in a structured manner. Because of the dependency on Evaluation of sub-activity (ADV_TDS.1)**,** the evaluator is expected to have identified the TSF prior to beginning work on this sub-activity. Without firm knowledge of what comprises the TSF, it is not possible to assess the completeness of the TSFI.

In performing the various work units included in this family, the evaluator is asked to make assessments of accuracy and completeness of several factors [the TSFI itself, as well as the individual components (parameters, actions, error messages, etc.) of the TSFI]. In doing this analysis, the evaluator is expected to use the documentation provided for the evaluation. This includes the ST, the TOE design, and may include other documentation such as the operational user guidance, security architecture description, and implementation representation. The documentation should be examined in an iterative fashion. The evaluator may read, for example, in the TOE design how a certain function is implemented, but see no way to invoke that function from the interface. This can cause the evaluator to question the completeness of a particular TSFI description, or whether an interface has been left out of the functional specification altogether. Describing analysis activities of this sort in the ETR is a key method in providing rationale that the work units have been performed appropriately.

It should be recognised that there exist functional requirements whose functionality is manifested wholly or in part architecturally, rather than through a specific mechanism. An example of this is the implementation of mechanisms implementing the Residual information protection (FDP_RIP) requirements. Such mechanisms typically are implemented to ensure a behaviour isn't present, which is difficult to test and typically is verified through analysis. In the cases where such functional requirements are included in the ST, it is expected that the evaluator recognise that there may be SFRs of this type that have no interfaces, and that this should not be considered a deficiency in the functional specification.

13.4.4.4 Action ADV_FSP.4.1E

13.4.4.4.1 General

CC Part 3 ADV_FSP.4.1C: *The functional specification shall completely represent the TSF.*

13.4.4.4.2 Work unit ADV_FSP.4-1

The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

The identification of the TSFI is a necessary prerequisite to all other activities in this sub-activity. The TSF must be identified (done as part of the TOE design (ADV_TDS) work units) in order to identify the TSFI. This activity can be done at a high level to ensure that no large groups of interfaces have been missed (network protocols, hardware interfaces, configuration files), or at a low level as the evaluation of the functional specification proceeds.

In making an assessment for this work unit, the evaluator determines that all portions of the TSF are addressed in terms of the interfaces listed in the functional specification. All portions of the TSF should have a corresponding interface description, or if there are no corresponding interfaces for a portion of the TSF, the evaluator determines that that is acceptable.

CC Part 3 ADV_FSP.4.2C: *The functional specification shall describe the purpose and method of use for all TSFI.*

13.4.4.4.3 Work unit ADV_FSP.4-2

The evaluator ***shall examine*** the functional specification to determine that it states the purpose of each TSFI.

The purpose of a TSFI is a general statement summarising the functionality provided by the interface. It is not intended to be a complete statement of the actions and results related to the interface, but rather a statement to help the reader understand in general what the interface is intended to be used for. The evaluator should not only determine that the purpose exists, but also that it accurately reflects the TSFI by taking into account other information about the interface, such as the description of actions and error messages.

13.4.4.4.4 Work unit ADV_FSP.4-3

The evaluator ***shall examine*** the functional specification to determine that the method of use for each TSFI is given.

The method of use for a TSFI summarizes how the interface is manipulated in order to invoke the actions and obtain the results associated with the TSFI. The evaluator should be able to determine, from reading this material in the functional specification, how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each TSFI, as it may be possible to describe in general how kernel calls are invoked, for instance, and then identify each interface using that general style. Different types of interfaces will require different method of use specifications. APIs, network protocol interfaces, system configuration parameters, and hardware bus interfaces all have very different methods of use, and this should be taken into account by the developer when developing the functional specification, as well as by the evaluator evaluating the functional specification.

For administrative interfaces whose functionality is documented as being inaccessible to untrusted users, the evaluator ensures that the method of making the functions inaccessible is described in the functional specification. It should be noted that this inaccessibility needs to be tested by the developer in their test suite.

The evaluator should not only determine that the set of method of use descriptions exist, but also that they accurately cover each TSFI.

13.4.4.4.5 Work unit ADV_FSP.4-4

The evaluator ***shall examine*** the functional specification to determine the completeness of the TSFI

The evaluator shall use the design documentation to identify the possible types of interfaces. The evaluator shall search the design documentation and the guidance documentation for potential TSFI not contained in the developer's documentation, thus indicating that the set of TSFI defined by the developer is incomplete. The evaluator ***shall examine*** the arguments presented by the developer that the TSFI is complete and check down to the lowest level of design or with the implementation representation that no additional TSFI exist.

CC Part 3 ADV_FSP.4.3C: *The functional specification shall identify and describe all parameters associated with each TSFI.*

13.4.4.4.6 Work unit ADV_FSP.4-5

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely identifies all parameters associated with every TSFI.

The evaluator examines the functional specification to ensure that all of the parameters are described for each TSFI. Parameters are explicit inputs or outputs to an interface that control the behaviour of that interface. For examples, parameters are the arguments supplied to an API; the various fields in packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

In order to determine that all of the parameters are present in the TSFI, the evaluator should examine the rest of the interface description (actions, error messages, etc.) to determine if the effects of the parameter are accounted for in the description. The evaluator should also check other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

13.4.4.4.7 Work unit ADV_FSP.4-6

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all parameters associated with every TSFI.

Once all of the parameters have been identified, the evaluator needs to ensure that they are accurately described, and that the description of the parameters is complete. A parameter description tells what the parameter is in some meaningful way. For instance, the interface *foo(i)* can be described as having "parameter i which is an integer"; this is not an acceptable parameter description. A description such as "parameter i is an integer that indicates the number of users currently logged in to the system" is much more acceptable.

In order to determine that the description of the parameters is complete, the evaluator should examine the rest of the interface description (purpose, method of use, actions, error messages, etc.) to determine if the descriptions of the parameter(s) are accounted for in the description. The evaluator should also check other evidence provided (e.g., TOE design, architectural design, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

CC Part 3 ADV_FSP.4.4C: *The functional specification shall describe all actions associated with each TSFI.*

13.4.4.4.8 Work unit ADV_FSP.4-7

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all actions associated with every TSFI.

The evaluator checks to ensure that all of the actions are described. actions available through an interface describe what the interface does (as opposed to the TOE design, which describes how the actions are provided by the TSF).

Actions of an interface describe functionality that can be invoked through the interface, and can be categorised as *regular* actions, and *SFR-related* actions. Regular actions are descriptions of what the interface does. The amount of information provided for this description is dependent on the complexity of the interface. The SFR-related actions are those that are visible at any external interface (for instance, audit activity caused by the invocation of an interface (assuming audit requirements are included in the ST) should be described, even though the result of that action is generally not visible through the invoked interface). Depending on the parameters of an interface, there may be many different actions able to be invoked through the interface (for instance, an API can have the first parameter be a "subcommand", and the following parameters be specific to that subcommand. The IOCTL API in some Unix systems is an example of such an interface).

In order to determine that the description of the actions of a TSFI is complete, the evaluator should review the rest of the interface description (parameter descriptions, error messages, etc.)

to determine if the actions described are accounted for. The evaluator should also analyse other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if there is evidence of actions that are described there but not in the functional specification.

CC Part 3 ADV_FSP.4.5C: *The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.*

13.4.4.4.9 Work unit ADV_FSP.4-8

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all error messages resulting from an invocation of each TSFI.

Errors can take many forms, depending on the interface being described. For an API, the interface itself may return an error code; set a global error condition, or set a certain parameter with an error code. For a configuration file, an incorrectly configured parameter may cause an error message to be written to a log file. For a hardware PCI card, an error condition may raise a signal on the bus, or trigger an exception condition to the CPU.

Errors (and the associated error messages) come about through the invocation of an interface. The processing that occurs in response to the interface invocation may encounter error conditions, which trigger (through an implementation-specific mechanism) an error message to be generated. In some instances this may be a return value from the interface itself; in other instances a global value may be set and checked after the invocation of an interface. It is likely that a TOE will have a number of low-level error messages that may result from fundamental resource conditions, such as "disk full" or "resource locked". While these error messages may map to a large number of TSFI, they can be used to detect instances where detail from an interface description has been omitted. For instance, a TSFI that produces a "disk full" message, but has no obvious description of why that TSFI should cause an access to the disk in its description of actions, can cause the evaluator to examine other evidence (Security Architecture (ADV_ARC)**,** TOE design (TOE_TDS)) related that TSFI to determine if the description is complete and accurate.

The evaluator determines that, for each TSFI, the exact set of error messages that can be returned on invoking that interface can be determined. The evaluator reviews the evidence provided for the interface to determine if the set of errors seems complete. They cross-check this information with other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to ensure that there are no errors steaming from processing mentioned that are not included in the functional specification.

13.4.4.4.10 Work unit ADV_FSP.4-9

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes the meaning of all error messages resulting from an invocation of each TSFI.

In order to determine accuracy, the evaluator must be able to understand meaning of the error. For example, if an interface returns a numeric code of 0, 1, or 2, the evaluator would not be able to understand the error if the functional specification only listed: "possible errors resulting from invocation of the *foo()* interface are 0, 1, or 2". Instead the evaluator checks to ensure that the errors are described such as: "possible errors resulting from invocation of the *foo()* interface are 0 (processing successful), 1 (file not found), or 2 (incorrect filename specification)".

In order to determine that the description of the errors due to invoking a TSFI is complete, the evaluator examines the rest of the interface description (parameter descriptions, actions, etc.) to determine if potential error conditions that can be caused by using such an interface are accounted for. The evaluator also checks other evidence provided for the evaluation (e.g. TOE

design, security architecture description, operational user guidance, implementation representation) to see if error processing related to the TSFI is described there but is not described in the functional specification.

CC Part 3 ADV_FSP.4.6C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.4.4.11 Work unit ADV_FSP.4-10

The evaluator ***shall check*** that the tracing links the SFRs to the corresponding TSFIs.

The tracing is provided by the developer to serve as a guide to which SFRs are related to which TSFIs. This tracing can be as simple as a table; it is used as input to the evaluator for use in the following work units, in which the evaluator verifies its completeness and accuracy.

13.4.4.5 Action ADV_FSP.4.2E

13.4.4.5.1 Work unit ADV_FSP.4-11

The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the SFRs.

To ensure that all SFRs are covered by the functional specification, as well as the test coverage analysis, the evaluator may build upon the developer's tracing (see ADV_FSP.4-10 a map between the TOE security functional requirements and the TSFI. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 assignment, and these ten rules were covered by three different TSFI, it would be inadequate for the evaluator to map FDP_ACC.1 to TSFI A, B, and C and claim they had completed the work unit. Instead, the evaluator would map FDP_ACC.1 (rule 1) to TSFI A; FDP_ACC.1 (rule 2) to TSFI B; etc. It can also be the case that the interface is a wrapper interface (e.g., IOCTL), in which case the mapping would need to be specific to certain set of parameters for a given interface.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that they completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST. It is also important to note that since the parameters, actions, and error messages associated with TSFIs must be fully specified, the evaluator should be able to determine if all aspects of an SFR appear to be implemented at the interface level.

13.4.4.5.2 Work unit ADV_FSP.4-12

The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the SFRs.

For each functional requirement in the ST that results in effects visible at the TSF boundary, the information in the associated TSFI for that requirement specifies the required functionality described by the requirement. For example, if the ST contains a requirement for access control lists, and the only TSFI that map to that requirement specify functionality for Unix-style protection bits, then the functional specification is not accurate with respect to the requirements.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that the evaluator completely map those requirements

to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST.

13.4.5 Evaluation of sub-activity (ADV_FSP.5)

13.4.5.1 Objectives

The objective of this sub-activity is to determine whether the developer has completely described all of the TSFI in a manner such that the evaluator is able to determine whether the TSFI are completely and accurately described, and appears to implement the security functional requirements of the ST. The completeness of the interfaces is judged based upon the implementation representation.

13.4.5.2 Input

The evaluation evidence for this sub-activity that is required by the work-units is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the implementation representation.

The evaluation evidence for this sub-activity that is used if included in the ST for the TOE is:

a) the security architecture description;

b) the TSF internals description;

c) the formal security policy model;

d) the operational user guidance.

13.4.5.3 Action ADV_FSP.5.1E

13.4.5.3.1 General

CC Part 3 ADV_FSP.5.1C: *The functional specification shall completely represent the TSF.*

13.4.5.3.2 Work unit ADV_FSP.5-1

The evaluator ***shall examine*** the functional specification to determine that the TSF is fully represented.

The identification of the TSFI is a necessary prerequisite to all other activities in this sub-activity. The TSF must be identified (done as part of the TOE design (TOE_TDS) work units) in order to identify the TSFI. This activity can be done at a high level to ensure that no large groups of interfaces have been missed (network protocols, hardware interfaces, configuration files), or at a low level as the evaluation of the functional specification proceeds.

In making an assessment for this work unit, the evaluator determines that all portions of the TSF are addressed in terms of the interfaces listed in the functional specification. All portions of the TSF should have a corresponding interface description, or if there are no corresponding interfaces for a portion of the TSF, the evaluator determines that that is acceptable.

CC Part 3 ADV_FSP.5.2C: *The functional specification shall describe the TSFI using a semi-formal style.*

13.4.5.3.3 Work unit ADV_FSP.5-2

The evaluator ***shall examine*** the functional specification to determine that it is presented using a semiformal style.

A semi-formal presentation is characterised by a standardised format with a well-defined syntax that reduces ambiguity that may occur in informal presentations. Since the intent of the semi-formal format is to enhance the reader's ability to understand the presentation, use of certain structured presentation methods (pseudo-code, flow charts, block diagrams) are appropriate, though not required.

For the purposes of this activity, the evaluator should ensure that the interface descriptions are formatted in a structured, consistent manner and use common terminology. A semiformal presentation of the interfaces also implies that the level of detail of the presentation for the interfaces is largely consistent across all TSFI. For the functional specification, it is acceptable to refer to external specifications for portions of the interface as long as those external specifications are themselves semiformal.

CC Part 3 ADV_FSP.5.3C: *The functional specification shall describe the purpose and method of use for all TSFI.*

13.4.5.3.4 Work unit ADV_FSP.5-3

The evaluator ***shall examine*** the functional specification to determine that it states the purpose of each TSFI.

The purpose of a TSFI is a general statement summarising the functionality provided by the interface. It is not intended to be a complete statement of the actions and results related to the interface, but rather a statement to help the reader understand in general what the interface is intended to be used for. The evaluator should not only determine that the purpose exists, but also that it accurately reflects the TSFI by taking into account other information about the interface, such as the description of actions and error messages.

13.4.5.3.5 Work unit ADV_FSP.5-4

The evaluator ***shall examine*** the functional specification to determine that the method of use for each TSFI is given.

The method of use for a TSFI summarizes how the interface is manipulated in order to invoke the actions and obtain the results associated with the TSFI. The evaluator should be able to determine, from reading this material in the functional specification, how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each TSFI, as it may be possible to describe in general how kernel calls are invoked, for instance, and then identify each interface using that general style. Different types of interfaces will require different method of use specifications. APIs, network protocol interfaces, system configuration parameters, and hardware bus interfaces all have very different methods of use, and this should be taken into account by the developer when developing the functional specification, as well as by the evaluator evaluating the functional specification.

For administrative interfaces whose functionality is documented as being inaccessible to untrusted users, the evaluator ensures that the method of making the functions inaccessible is described in the functional specification. It should be noted that this inaccessibility needs to be tested by the developer in their test suite.

The evaluator should not only determine that the set of method of use descriptions exist, but also that they accurately cover each TSFI.

13.4.5.3.6 Work unit ADV_FSP.5-5

The evaluator ***shall examine*** the functional specification to determine the completeness of the TSFI

The evaluator shall use the design documentation to identify the possible types of interfaces. The evaluator shall search the design documentation and the guidance documentation for potential TSFI not contained in the developer's documentation, thus indicating that the set of TSFI defined by the developer is incomplete. The evaluator ***shall examine*** the arguments presented by the developer that the TSFI is complete and check down to the lowest level of design or with the implementation representation that no additional TSFI exist.

CC Part 3 ADV_FSP.5.4C: *The functional specification shall identify and describe all parameters associated with each TSFI.*

13.4.5.3.7 Work unit ADV_FSP.5-6

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely identifies all parameters associated with every TSFI.

The evaluator examines the functional specification to ensure that all of the parameters are described for each TSFI. Parameters are explicit inputs or outputs to an interface that control the behaviour of that interface. For examples, parameters are the arguments supplied to an API; the various fields in packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

In order to determine that all of the parameters are present in the TSFI, the evaluator should examine the rest of the interface description (actions, error messages, etc.) to determine if the effects of the parameter are accounted for in the description. The evaluator should also check other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

13.4.5.3.8 Work unit ADV_FSP.5-7

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all parameters associated with every TSFI.

Once all of the parameters have been identified, the evaluator needs to ensure that they are accurately described, and that the description of the parameters is complete. A parameter description tells what the parameter is in some meaningful way. For instance, the interface *foo(i)* can be described as having "parameter i which is an integer"; this is not an acceptable parameter description. A description such as "parameter i is an integer that indicates the number of users currently logged in to the system". is much more acceptable.

In order to determine that the description of the parameters is complete, the evaluator should examine the rest of the interface description (purpose, method of use, actions, error messages, etc.) to determine if the descriptions of the parameter(s) are accounted for in the description. The evaluator should also check other evidence provided (e.g., TOE design, architectural design, operational user guidance, implementation representation) to see if behaviour or additional parameters are described there but not in the functional specification.

CC Part 3 ADV_FSP.5.5C: *The functional specification shall describe all actions associated with each TSFI.*

13.4.5.3.9 Work unit ADV_FSP.5-8

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all actions associated with every TSFI.

The evaluator checks to ensure that all of the actions are described. actions available through an interface describe what the interface does (as opposed to the TOE design, which describes how the actions are provided by the TSF).

actions of an interface describe functionality that can be invoked through the interface, and can be categorised as *regular* actions, and *SFR-related* actions. Regular actions are descriptions of what the interface does. The amount of information provided for this description is dependent on the complexity of the interface. The SFR-related actions are those that are visible at any external interface (for instance, audit activity caused by the invocation of an interface (assuming audit requirements are included in the ST) should be described, even though the result of that action is generally not visible through the invoked interface). Depending on the parameters of an interface, there may be many different actions able to be invoked through the interface (for instance, an API might have the first parameter be a "subcommand", and the following parameters be specific to that subcommand. The IOCTL API in some Unix systems is an example of such an interface).

In order to determine that the description of the actions of a TSFI is complete, the evaluator should review the rest of the interface description (parameter descriptions, error messages, etc.) to determine if the actions described are accounted for. The evaluator should also analyse other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if there is evidence of actions that are described there but not in the functional specification.

CC Part 3 ADV_FSP.5.6C: *The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.*

13.4.5.3.10 Work unit ADV_FSP.5-9

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes all error messages resulting from an invocation of each TSFI.

Errors can take many forms, depending on the interface being described. For an API, the interface itself may return an error code; set a global error condition, or set a certain parameter with an error code. For a configuration file, an incorrectly configured parameter may cause an error message to be written to a log file. For a hardware PCI card, an error condition may raise a signal on the bus, or trigger an exception condition to the CPU.

Errors (and the associated error messages) come about through the invocation of an interface. The processing that occurs in response to the interface invocation may encounter error conditions, which trigger (through an implementation-specific mechanism) an error message to be generated. In some instances this may be a return value from the interface itself; in other instances a global value may be set and checked after the invocation of an interface. It is likely that a TOE will have a number of low-level error messages that may result from fundamental resource conditions, such as "disk full" or "resource locked". While these error messages may map to a large number of TSFI, they can be used to detect instances where detail from an interface description has been omitted. For instance, a TSFI that produces a "disk full" message, but has no obvious description of why that TSFI should cause an access to the disk in its description of actions, can cause the evaluator to examine other evidence (ADV_ARC, ADV_TDS) related that TSFI to determine if the description is complete and accurate.

The evaluator determines that, for each TSFI, the exact set of error messages that can be returned on invoking that interface can be determined. The evaluator reviews the evidence provided for

the interface to determine if the set of errors seems complete. They cross-check this information with other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to ensure that there are no errors steaming from processing mentioned that are not included in the functional specification.

13.4.5.3.11 Work unit ADV_FSP.5-10

The evaluator ***shall examine*** the presentation of the TSFI to determine that it completely and accurately describes the meaning of all error messages resulting from an invocation of each TSFI.

In order to determine accuracy, the evaluator must be able to understand meaning of the error. For example, if an interface returns a numeric code of 0, 1, or 2, the evaluator would not be able to understand the error if the functional specification only listed: "possible errors resulting from invocation of the *foo()* interface are 0, 1, or 2". Instead the evaluator checks to ensure that the errors are described such as: "possible errors resulting from invocation of the *foo()* interface are 0 (processing successful), 1 (file not found), or 2 (incorrect filename specification)".

In order to determine that the description of the errors due to invoking a TSFI is complete, the evaluator examines the rest of the interface description (parameter descriptions, actions, etc.) to determine if potential error conditions that can be caused by using such an interface are accounted for. The evaluator also checks other evidence provided for the evaluation (e.g., TOE design, security architecture description, operational user guidance, implementation representation) to see if error processing related to the TSFI is described there but is not described in the functional specification.

CC Part 3 ADV_FSP.5.7C: *The functional specification shall describe all error messages that do not result from an invocation of a TSFI.*

13.4.5.3.12 Work unit ADV_FSP.5-11

The evaluator ***shall examine*** the functional specification to determine that it completely and accurately describes all error messages that do not result from an invocation of any TSFI.

This work unit complements work unit ADV_FSP.5-9, which describes those error messages that result from an invocation of the TSFI. Taken together, these work units cover all error messages that can be generated by the TSF.

The evaluator assesses the completeness and accuracy of the functional specification by comparing its contents to instances of error message generation within the implementation representation. Most of these error messages will have already been covered by work unit ADV_FSP.5-9.

The error messages related to this work unit are typically those that are not expected to be generated, but are constructed as a matter of good programming practises. For example, a case statement that defines actions resulting from each of a list of cases may end with a final *else* statement to apply to anything that cannot be expected; this practise ensures the TSF does not get into an undefined state. However, it is not expected that the path of execution would ever get to this *else* statement; therefore, any error message generation within this *else* statement would never be generated. Although it would not get generated, it must still be included in the functional specification.

CC Part 3 ADV_FSP.5.8C: *The functional specification shall provide a rationale for each error message contained in the TSF implementation yet does not result from an invocation of a TSFI.*

13.4.5.3.13 Work unit ADV_FSP.5-12

The evaluator ***shall examine*** the functional specification to determine that it provides a rationale for each error message contained in the TSF implementation yet does not result from an invocation of a TSFI.

The evaluator ensures that every error message found under work unit ADV_FSP.5-11 contains a rationale describing why it cannot be invoked from the TSFI.

As was described in the previous work unit, this rationale can be as straightforward as the fact that the error message in question is provided for completeness of execution logic and that it is never expected to be generated. The evaluator ensures that the rationale for each such error message is logical.

CC Part 3 ADV_FSP.5.9C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.5.3.14 Work unit ADV_FSP.5-13

The evaluator ***shall check*** that the tracing links the SFRs to the corresponding TSFIs.

The tracing is provided by the developer to serve as a guide to which SFRs are related to which TSFIs. This tracing can be as simple as a table; it is used as input to the evaluator for use in the following work units, in which the evaluator verifies its completeness and accuracy.

13.4.5.4 Action ADV_FSP.5.2E

13.4.5.4.1 Work unit ADV_FSP.5-14

The evaluator ***shall examine*** the functional specification to determine that it is a complete instantiation of the SFRs.

To ensure that all SFRs are covered by the functional specification, as well as the test coverage analysis, the evaluator may build upon the developer's tracing (see ADV_FSP.5-13 a map between the TOE security functional requirements and the TSFI. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 assignment, and these ten rules were covered by three different TSFI, it would be inadequate for the evaluator to map FDP_ACC.1 to TSFI A, B, and C and claim they had completed the work unit. Instead, the evaluator would map FDP_ACC.1 (rule 1) to TSFI A; FDP_ACC.1 (rule 2) to TSFI B; etc. It can also be the case that the interface is a wrapper interface (e.g., IOCTL), in which case the mapping would need to be specific to certain set of parameters for a given interface.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that they completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST. It is also important to note that since the parameters, actions, and error messages associated with TSFIs must be fully specified, the evaluator should be able to determine if all aspects of an SFR appear to be implemented at the interface level.

13.4.5.4.2 Work unit ADV_FSP.5-15

The evaluator ***shall examine*** the functional specification to determine that it is an accurate instantiation of the SFRs.

For each functional requirement in the ST that results in effects visible at the TSF boundary, the information in the associated TSFI for that requirement specifies the required functionality described by the requirement. For example, if the ST contains a requirement for access control lists, and the only TSFI that map to that requirement specify functionality for Unix-style protection bits, then the functional specification is not accurate with respect to the requirements.

The evaluator must recognise that for requirements that have little or no manifestation at the TSF boundary (e.g., FDP_RIP) it is not expected that the evaluator completely map those requirements to the TSFI. The analysis for those requirements will be performed in the analysis for the TOE design (ADV_TDS) when included in the ST.

CC Part 3 ADV_FSP.6.1C: *The functional specification shall completely represent the TSF.*

CC Part 3 ADV_FSP.6.2C: *The functional specification shall describe the TSFI using a **formal** style.*

CC Part 3 ADV_FSP.6.3C: *The functional specification shall describe the purpose and method of use for all TSFI.*

CC Part 3 ADV_FSP.6.4C: *The functional specification shall identify and describe all parameters associated with each TSFI.*

CC Part 3 ADV_FSP.6.5C: *The functional specification shall describe all actions associated with each TSFI.*

CC Part 3 ADV_FSP.6.6C: *The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.*

CC Part 3 ADV_FSP.6.7C: *The functional specification shall describe all error messages contained in the TSF implementation representation.*

CC Part 3 ADV_FSP.6.8C: *The functional specification shall provide a rationale for each error message contained in the TSF implementation that is not otherwise described in the functional specification justifying why it is not associated with a TSFI.*

CC Part 3 ADV_FSP.6.9C: *The formal presentation of the functional specification of the TSF shall describe the TSFI using a formal style, supported by informal, explanatory text where appropriate.*

CC Part 3 ADV_FSP.6.10C: *The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.*

13.4.6 Evaluation of sub-activity (ADV_FSP.6)

There is no general guidance; the scheme should be consulted for guidance on this sub-activity.

13.5 Implementation representation (ADV_IMP)

13.5.1 Evaluation of sub-activity (ADV_IMP.1)

13.5.1.1 Objectives

The objective of this sub-activity is to determine that the implementation representation made available by the developer is suitable for use in other analysis activities; *suitability* is judged by its conformance to the requirements for this component.

13.5.1.2 Input

The evaluation evidence for this sub-activity is:

a) the implementation representation;

b) the documentation of the development tools, as resulting from ALC_TAT ;

c) TOE design description.

13.5.1.3 Application notes

The entire implementation representation is made available to ensure that analysis activities are not curtailed due to lack of information. This does not, however, imply that all of the representation is examined when the analysis activities are being performed. This is likely impractical in almost all cases, in addition to the fact that it most likely will not result in a higher-assurance TOE vs. targeted sampling of the implementation representation. For this sub-activity, this is even truer. It would not be productive for the evaluator to spend large amounts of time verifying the requirements for one portion of the implementation representation, and then use a different portion of the implementation representation in performing analysis for other work units. Therefore, the evaluator is encouraged to select the sample of the implementation representation from the areas of the TOE that will be of most interest during the analysis performed during work units from other families (e.g. ATE_IND, AVA_VAN and ADV_INT).

13.5.1.4 Action ADV_IMP.1.1E

13.5.1.4.1 General

CC Part 3 ADV_IMP.1.1C: *The implementation representation shall define the TSF to a level of detail such that the TSF may be generated without further design decisions.*

13.5.1.4.2 Work unit ADV_IMP.1-1

The evaluator ***shall check*** that the implementation representation defines the TSF to a level of detail such that the TSF can be generated without further design decisions.

Source code or hardware diagrams and/or IC hardware design language code or layout data that are used to build the actual hardware are examples of parts of an implementation representation. The evaluator samples the implementation representation to gain confidence that it is at the appropriate level and not, for instance, a pseudo-code level which requires additional design decisions to be made. The evaluator is encouraged to perform a quick check when first looking at the implementation representation to assure themselves that the developer has supplied all the required information. However, the evaluator is also encouraged to perform the bulk of this check while working on other work units that call for examining the implementation; this will ensure the sample examined for this work unit is relevant.

CC Part 3 ADV_IMP.1.2C: *The implementation representation shall be in the form used by the development personnel.*

13.5.1.4.3 Work unit ADV_IMP.1-2

The evaluator ***shall check*** that the implementation representation is in the form used by development personnel.

The implementation representation is manipulated by the developer in a form that is suitable for transformation to the actual implementation. For instance, the developer may work with files containing source code, which is eventually compiled to become part of the TSF. The developer makes available the implementation representation in the form they use, so that the evaluator may use automated techniques in the analysis. This also increases the confidence that the implementation representation examined is actually the one used in the production of the TSF (as opposed to the case where it is supplied in an alternate presentation format, such as a word processor document). It should be noted that other forms of the implementation representation

may also be used by the developer; these forms are supplied as well. The overall goal is to supply the evaluator with the information that will maximise the evaluator's analysis efforts.

The evaluator samples the implementation representation to gain confidence that it is the version that is usable by the developer. The sample is such that the evaluator has assurance that all areas of the implementation representation are in conformance with the requirement; however, a complete examination of the entire implementation representation is unnecessary.

Conventions in some forms of the implementation representation may make it difficult or impossible to determine from just the implementation representation itself what the actual result of the compilation or run-time interpretation will be. For example, compiler directives for C language compilers will cause the compiler to exclude or include entire portions of the code.

Some forms of the implementation representation may require additional information because they introduce significant barriers to understanding and analysis. Examples include shrouded source code or source code that has been obfuscated in other ways such that it prevents understanding and/or analysis. These forms of implementation representation typically result from by taking a version of the implementation representation that is used by the TOE developer and running a shrouding or obfuscation program on it. While the shrouded representation is what is compiled and may be closer to the implementation (in terms of structure) than the original, un-shrouded representation, supplying such obfuscated code may cause significantly more time to be spent in analysis tasks involving the representation. When such forms of representation are created, the components require details on the shrouding tools/algorithms used so that the un-shrouded representation can be supplied, and the additional information can be used to gain confidence that the shrouding process does not compromise any security mechanisms.

The evaluator samples the implementation representation to gain confidence that all of the information needed to interpret the implementation representation has been supplied. Note that the tools are among those referenced by Tools and techniques (ALC_TAT) components. The evaluator is encouraged to perform a quick check when first looking at the implementation representation to assure themselves that the developer is on the right track. However, the evaluator is also encouraged to perform the bulk of this check while working on other work units that call for examining the implementation; this will ensure the sample examined for this work unit is relevant.

CC Part 3 ADV_IMP.1.3C: *The mapping between the TOE design description and the sample of the implementation representation shall demonstrate their correspondence.*

13.5.1.4.4 Work unit ADV_IMP.1-3

The evaluator ***shall examine*** the mapping between the TOE design description and the sample of the implementation representation to determine that it is accurate.

The evaluator augments the determination of existence (specified in work unit ADV_IMP.1-1) by verifying the accuracy of a portion of the implementation representation and the TOE design description. For parts of the TOE design description that are interesting, the evaluator would verify the implementation representation accurately reflects the description provided in the TOE design description.

For example, the TOE design description can identify a login module that is used to identify and authenticate users. If user authentication is sufficiently significant, the evaluator would verify that the corresponding code in fact implements that service as described in the TOE design description. It can also be worthwhile to verify that the code accepts the parameters as described in the functional specification.

It is worth pointing out the developer must choose whether to perform the mapping for the entire implementation representation, thereby guaranteeing that the chosen sample will be covered, or

waiting for the sample to be chosen before performing the mapping. The first option is likely more work, but may be completed before the evaluation begins. The second option is less work, but will produce a suspension of evaluation activity while the necessary evidence is being produced.

13.5.2 Evaluation of sub-activity (ADV_IMP.2)

13.5.2.1 Objectives

The objective of this sub-activity is to determine that the implementation representation made available by the developer is suitable for use in other analysis activities; suitability is judged by its conformance to the requirements for this component.

13.5.2.2 Input

The evaluation evidence for this sub-activity is:

a) the implementation representation;

b) the documentation of the development tools, as resulting from ALC_TAT;

c) the TOE design description.

13.5.2.3 Application notes

The entire implementation representation is made available to ensure that analysis activities are not curtailed due to lack of information. This does not, however, imply that all of the representation is examined in detail when the analysis activities are being performed. This is likely impractical in almost all cases, in addition to the fact that it most likely will not result in a higher-assurance TOE.

The new aspect for ADV_IMP.2 in comparison to ADV_IMP.1 is that the developer needs to demonstrate and the evaluator will confirm that the complete implementation representation is mapped to the TOE design description. This does, however, not imply that all other work units need an examination of the complete implementation representation. Aspects like appropriate level of detail and form of the implementation representation can be covered by sampling as for ADV_IMP.1.

13.5.2.4 Action ADV_IMP.2.1E

13.5.2.4.1 General

CC Part 3 ADV_IMP.2.1C: *The implementation representation shall define the TSF to a level of detail such that the TSF may be generated without further design decisions.*

13.5.2.4.2 Work unit ADV_IMP.2-1

The evaluator ***shall check*** that the implementation representation defines the TSF to a level of detail such that the TSF can be generated without further design decisions.

Source code or hardware diagrams and/or IC hardware design language code or layout data that are used to build the actual hardware are examples of parts of an implementation representation. The evaluator samples the implementation representation to gain confidence that it is at the appropriate level and not, for instance, a pseudo-code level which requires additional design decisions to be made. The evaluator is encouraged to perform a quick check when first looking at the implementation representation to assure themselves that the developer is on the right track. However, the evaluator is also encourage to perform the bulk of this check while working on other work units that call for examining the implementation; this will ensure the sample examined for this work unit is relevant.

If the evaluator has the possibility to actually execute or witness the "built" procedure used to transfer the implementation representation into the actual implementation, and to compare the result to the TOE as delivered, this may provide an easier and at the same time more reliable check for this work unit (and possibly also for the following one).

CC Part 3 ADV_IMP.2.2C: *The implementation representation shall be in the form used by the development personnel.*

13.5.2.4.3 Work unit ADV_IMP.2-2

The evaluator ***shall check*** that the implementation representation is in the form used by development personnel.

The implementation representation is manipulated by the developer in a form that is suitable for transformation to the actual implementation. For instance, the developer may work with files containing source code, which is eventually compiled to become part of the TSF. The developer makes available the implementation representation in the form they use, so that the evaluator may use automated techniques in the analysis. This also increases the confidence that the implementation representation examined is actually the one used in the production of the TSF (as opposed to the case where it is supplied in an alternate presentation format, such as a word processor document). It should be noted that other forms of the implementation representation may also be used by the developer; these forms are supplied as well. The overall goal is to supply the evaluator with the information that will maximise the evaluator's analysis efforts.

The evaluator samples the implementation representation to gain confidence that it is the version that is usable by the developer. The sample is such that the evaluator has assurance that all areas of the implementation representation are in conformance with the requirement; however, a complete examination of the entire implementation representation is unnecessary.

Conventions in some forms of the implementation representation may make it difficult or impossible to determine from just the implementation representation itself what the actual result of the compilation or run-time interpretation will be. For example, compiler directives for C language compilers will cause the compiler to exclude or include entire portions of the code.

Some forms of the implementation representation may require additional information because they introduce significant barriers to understanding and analysis. Examples include shrouded source code or source code that has been obfuscated in other ways such that it prevents understanding and/or analysis. These forms of implementation representation typically result from by taking a version of the implementation representation that is used by the TOE developer and running a shrouding or obfuscation program on it. While the shrouded representation is what is compiled and may be closer to the implementation (in terms of structure) than the original, un-shrouded representation, supplying such obfuscated code may cause significantly more time to be spent in analysis tasks involving the representation. When such forms of representation are created, the components require details on the shrouding tools/algorithms used so that the un-shrouded representation can be supplied, and the additional information can be used to gain confidence that the shrouding process does not compromise any security mechanisms.

The evaluator samples the implementation representation to gain confidence that all of the information needed to interpret the implementation representation has been supplied. Note that the tools are among those referenced by Tools and techniques (ALC_TAT) components. The evaluator is encouraged to perform a quick check when first looking at the implementation representation to assure themselves that the developer is on the right track. However, the evaluator is also encouraged to perform the bulk of this check while working on other work units that call for examining the implementation; this will ensure the sample examined for this work unit is relevant.

CC Part 3 ADV_IMP.2.3C: *The mapping between the TOE design description and the entire implementation representation shall demonstrate their correspondence.*

13.5.2.4.4 Work unit ADV_IMP.2-3

The evaluator ***shall examine*** the mapping between the TOE design description and the entire implementation representation to determine that it is accurate.

The evaluator augments the determination of existence (specified in work unit ADV_IMP.2-1) by verifying the accuracy of the implementation representation and the TOE design description. For those parts of TOE design description that are interesting, the evaluator would verify the implementation representation accurately reflects the description provided in the TOE design description.

For example, the TOE design description can identify a login module that is used to identify and authenticate users. If user authentication is sufficiently significant, the evaluator would verify that the corresponding code in fact implements that service as described in the TOE design description. It can also be worthwhile to verify that the code accepts the parameters as described in the functional specification.

Usually it will be expected that the evaluator considers at least the functionality required by the SFRs chosen in the ST and aspects described in the security architecture description as "interesting" in the sense discussed above. Note however that not all aspects of the security architecture are necessarily traceable to specific parts of the implementation representation.

It is worth pointing out the developer must perform the mapping for the entire implementation representation, thereby guaranteeing that the chosen sample will be covered.

13.5.2.4.5 Work unit ADV_IMP.2-4

The evaluator ***shall examine*** the mapping between the TOE design description and the entire implementation representation to determine that it is complete.

Note that the completeness here is relevant in both directions: The complete TOE design needs to be covered by the implementation representation and all parts of the implementation representation needs to be mapped to a corresponding part of the TOE design.

In order to confirm that the entire implementation representation is covered by the mapping the evaluator will not need to examine the content of every part of the implementation representation. If (in the case of a software TOE) the mapping is for example described by mapping each source code file to a module in the TOE design description, it will be sufficient if this mapping is plausible from the role of the source code file the evaluator can conclude from information like the naming of the source code files, their grouping in subdirectories or their grouping in "built" procedures. Note, that aspects of accuracy are covered by the preceding work unit.

In order to confirm that the entire design description is covered by the implementation, the evaluator may either use a similar argument as in the other direction, i. e. that all modules contained in the TOE design description are mapped to parts of the implementation representation in a plausible way. In addition, if the evaluator has established in the preceding work unit that all SFRs and all applicable parts of the security architecture description are traceable to the implementation representation this may be seen as sufficient evidence that the mapping is complete.

13.6 TSF internals (ADV_INT)

13.6.1 Evaluation of sub-activity (ADV_INT.1)

13.6.1.1 Objectives

The objective of this sub-activity is to determine whether the defined subset of the TSF is designed and structured such that the likelihood of flaws is reduced and that maintenance can be more readily performed without the introduction of flaws.

13.6.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE design description;

c) the implementation representation (if ADV_IMP is part of the claimed assurance);

d) the TSF internals description and justification;

e) the documentation of the coding standards, as resulting from ALC_TAT.

13.6.1.3 Application notes

The role of the internals description is to provide evidence of the structure of the design and implementation of the TSF.

The structure of the design has two aspects: the constituent parts of the TSF and the procedures used to design the TSF. In cases where the TSF is designed in a manner consistent with the design represented by the TOE design (see ADV_TDS), the assessment of the TSF design is obvious. In cases where the design procedures (see ALC_TAT) are being followed, the assessment of the TSF design procedures is similarly obvious.

In cases where the TSF is implemented using procedure-based software, this structure is assessed on the basis of its modularity; the modules identified in the internals description are the same as the modules identified in the TOE design (TOE design (TOE_TDS)). A module consists of one or more source code files that cannot be decomposed into smaller compilable units.

The use of the assignment in this component levies stricter constraints on the subset of the TSF that is explicitly identified in the assignment ADV_INT.1.1D than on the remainder of the TSF. While the entire TSF is to be designed using good engineering principles and result in a well-structured TSF, only the specified subset is specifically analysed for this characteristic. The evaluator determines that the developer's application of coding standards result in a TSF that is understandable.

The primary goal of this component is to ensure the TSF subset's implementation representation is understandable to facilitate maintenance and analysis (of both the developer and evaluator).

13.6.1.4 Action ADV_INT.1.1E

13.6.1.4.1 General

CC Part 3 ADV_INT.1.1C: *The justification shall explain the characteristics used to judge the meaning of "well-structured".*

13.6.1.4.2 Work unit ADV_INT.1-1

The evaluator ***shall examine*** the justification to determine that it identifies the basis for determining whether the TSF is well-structured.

The evaluator verifies that the criteria for determining the characteristic of being well-structured are clearly defined in the justification. Acceptable criteria typically originate from industry standards for the technology discipline. For example, procedural software that executes linearly is traditionally viewed as well-structured if it adheres to software engineering programming practises, such as those defined in the IEEE Standard (*IEEE Std 610.12-1990*). For example, it would identify the criteria for the procedural software portions of the TSF subset:

a) the process used for modular decomposition;

b) coding standards used in the development of the implementation;

c) a description of the maximum acceptable level of intermodule coupling exhibited by the TSF subset;

d) a description of the minimum acceptable level of cohesion exhibited the modules of the TSF subset.

For other types of technologies used in the TOE - such as non-procedural software (e.g. object-oriented programming), widespread commodity hardware (e.g. PC microprocessors), and special-purpose hardware (e.g. smart-card processors) - the evaluator should seek guidance from the evaluation authority for determining the adequacy of criteria for being "well-structured".

CC Part 3 ADV_INT.1.2C: *The TSF internals description shall demonstrate that the assigned subset of the TSF is well-structured.*

13.6.1.4.3 Work unit ADV_INT.1-2

The evaluator ***shall check*** the TSF internals description to determine that it identifies the assigned subset of the TSF.

This subset may be identified in terms of the internals of the TSF at any layer of abstraction. For example, it may be in terms of the structural elements of the TSF as identified in the TOE design (e.g. the audit subsystem), or in terms of the implementation (e.g. *encrypt.c* and *decrypt.c* files, or the 6227 IC chip).

It is insufficient to identify this subset in terms of the claimed SFRs (e.g. the portion of the TSF that provide anonymity as defined in FPR_ANO.2) because this does not indicate where to focus the analysis.

13.6.1.4.4 Work unit ADV_INT.1-3

The evaluator ***shall examine*** the TSF internals description to determine that it demonstrates that the assigned TSF subset is well-structured.

The evaluator examines the internals description to ensure that it provides a sound explanation of how the TSF subset meets the criteria from ADV_INT.1-1

For example, it would explain how the procedural software portions of the TSF subset meets the following:

a) that there is a one-to-one correspondence between the modules identified in the TSF subset and the modules described in the TOE design (ADV_TDS);

b) how the TSF design is a reflection of the modular decomposition process;

c) a justification for all instances where the coding standards were not used or met;

d) a justification for any coupling or cohesion outside the acceptable bounds.

13.6.1.5 Action ADV_INT.1.2E

13.6.1.5.1 Work unit ADV_INT.1-4

The evaluator ***shall determine*** that the TOE design for the assigned TSF subset is well-structured.

The evaluator examines a sample of the TOE design to verify the accuracy of the justification. For example, a sample of the TOE design is analysed to determine its adherence to the design standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope

The description of the TOE's decomposition into subsystems and modules will make the argument that the TSF subset is well-structured self-evident. Verification that the procedures for structuring the TSF (as examined in ALC_TAT) are being followed will make it self-evident that the TSF subset is well-structured.

13.6.1.5.2 Work unit ADV_INT.1-5

The evaluator ***shall determine*** that the assigned TSF subset is well-structured.

If ADV_IMP is not part of the claimed assurance, then this work unit is not applicable and is therefore considered to be satisfied.

The evaluator examines a sample of the TSF subset to verify the accuracy of the internals description. For example, a sample of the procedural software portions of the TSF subset is analysed to determine its cohesion and coupling, its adherence to the coding standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope.

13.6.2 Evaluation of sub-activity (ADV_INT.2)

13.6.2.1 Objectives

The objective of this sub-activity is to determine whether the TSF is designed and structured such that the likelihood of flaws is reduced and that maintenance can be more readily performed without the introduction of flaws.

13.6.2.2 Input

The evaluation evidence for this sub-activity is:

a) the modular design description;

b) the implementation representation (if ADV_IMP is part of the claimed assurance);

c) the TSF internals description;

d) the documentation of the coding standards, as resulting from ALC_TAT.

13.6.2.3 Application notes

The role of the internals description is to provide evidence of the structure of the design and implementation of the TSF.

The structure of the design has two aspects: the constituent parts of the TSF and the procedures used to design the TSF. In cases where the TSF is designed in a manner consistent with the design represented by the TOE design (see ADV_TDS), the assessment of the TSF design is obvious. In cases where the design procedures (see ALC_TAT) are being followed, the assessment of the TSF design procedures is similarly obvious.

In cases where the TSF is implemented using procedure-based software, this structure is assessed on the basis of its modularity; the modules identified in the internals description are the same as the modules identified in the TOE design (TOE design (ADV_TDS)). A module consists of one or more source code files that cannot be decomposed into smaller compilable units.

The primary goal of this component is to ensure the TSF's implementation representation is understandable to facilitate maintenance and analysis (of both the developer and evaluator).

13.6.2.4 Action ADV_INT.2.1E

13.6.2.4.1 General

CC Part 3 ADV_INT.2.1C: *The justification shall describe the characteristics used to judge the meaning of "well-structured".*

13.6.2.4.2 Work unit ADV_INT.2-1

The evaluator ***shall examine*** the justification to determine that it identifies the basis for determining whether the TSF is well-structured.

The evaluator verifies that the criteria for determining the characteristic of being well-structured are clearly defined in the justification. Acceptable criteria typically originate from industry standards for the technology discipline. For example, procedural software that executes linearly is traditionally viewed as well-structured if it adheres to software engineering programming practises, such as those defined in the IEEE Standard (*IEEE Std 610.12-1990*). For example, it would identify the criteria for the procedural software portions of the TSF:

a) the process used for modular decomposition;

b) coding standards used in the development of the implementation;

c) a description of the maximum acceptable level of intermodule coupling exhibited by the TSF;

d) a description of the minimum acceptable level of cohesion exhibited the modules of the TSF.

For other types of technologies used in the TOE - such as non-procedural software (e.g. object-oriented programming), widespread commodity hardware (e.g. PC microprocessors), and special-purpose hardware (e.g. smart-card processors) - the evaluation authority should be consulted for determining the adequacy of criteria for being "well-structured".

CC Part 3 ADV_INT.2.2C: *The TSF internals description shall demonstrate that the entire TSF is well-structured.*

13.6.2.4.3 Work unit ADV_INT.2-2

The evaluator ***shall examine*** the TSF internals description to determine that it demonstrates that the TSF is well-structured.

The evaluator examines the internals description to ensure that it provides a sound explanation of how the TSF meets the criteria from ADV_INT.2-1

For example, it would explain how the procedural software portions of the TSF meet the following:

a) that there is a one-to-one correspondence between the modules identified in the TSF and the modules described in the TOE design (ADV_TDS);

b) how the TSF design is a reflection of the modular decomposition process;

c) a justification for all instances where the coding standards were not used or met;

d) a justification for any coupling or cohesion outside the acceptable bounds.

13.6.2.5 Action ADV_INT.2.2E

13.6.2.5.1 Work unit ADV_INT.2-3

The evaluator ***shall determine*** that the TOE design is well-structured.

The evaluator examines the TOE design of a sample of the TSF to verify the accuracy of the justification. For example, a sample of the TOE design is analysed to determine its adherence to the design standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope

The description of the TOE's decomposition into subsystems and modules will make the argument that the TSF subset is well-structured self-evident. Verification that the procedures for structuring the TSF (as examined in ALC_TAT) are being followed will make it self-evident that the TSF subset is well-structured.

13.6.2.5.2 Work unit ADV_INT.2-4

The evaluator ***shall determine*** that the TSF is well-structured.

If ADV_IMP is not part of the claimed assurance, then this work unit is not applicable and is therefore considered to be satisfied.

The evaluator examines a sample of the TSF to verify the accuracy of the internals description. For example, a sample of the procedural software portions of the TSF is analysed to determine its cohesion and coupling, its adherence to the coding standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope.

13.6.3 Evaluation of sub-activity (ADV_INT.3)

13.6.3.1 Objectives

The objective of this sub-activity is to determine whether the TSF is designed and structured such that the likelihood of flaws is reduced and that maintenance can be more readily performed without the introduction of flaws.

13.6.3.2 Input

The evaluation evidence for this sub-activity is:

a) the modular design description;

b) the implementation representation (if ADV_IMP is part of the claimed assurance);

c) the TSF internals description;

d) the documentation of the coding standards, as resulting from ALC_TAT.

13.6.3.3 Application notes

The role of the internals description is to provide evidence of the structure of the design and implementation of the TSF.

The structure of the design has two aspects: the constituent parts of the TSF and the procedures used to design the TSF. In cases where the TSF is designed in a manner consistent with the design represented by the TOE design (see ADV_TDS), the assessment of the TSF design is obvious. In cases where the design procedures (see ALC_TAT) are being followed, the assessment of the TSF design procedures is similarly obvious.

In cases where the TSF is implemented using procedure-based software, this structure is assessed on the basis of its modularity; the modules identified in the internals description are the same as the modules identified in the TOE design (TOE design (ADV_TDS)). A module consists of one or more source code files that cannot be decomposed into smaller compilable units.

The primary goal of this component is to ensure the TSF's implementation representation is understandable to facilitate maintenance and analysis (of both the developer and evaluator).

13.6.3.4 Action ADV_INT.3.1E

13.6.3.4.1 General

CC Part 3 ADV_INT.3.1C: *The justification shall describe the characteristics used to judge the meaning of "well-structured" and "complex".*

13.6.3.4.2 Work unit ADV_INT.3-1

The evaluator ***shall examine*** the justification to determine that it identifies the basis for determining whether the TSF is "well-structured" and "not overly complex".

The evaluator verifies that the criteria for determining the characteristic of being "well-structured" and "complex" are clearly defined in the justification. Acceptable criteria typically originate from industry standards for the technology discipline. For example, procedural software that executes linearly is traditionally viewed as well-structured if it adheres to software engineering programming practises, such as those defined in the IEEE Standard (IEEE Std 610.12-1990). For example, it would identify the criteria for the procedural software portions of the TSF:

a) the process used for modular decomposition;

b) coding standards used in the development of the implementation;

c) a description of the maximum acceptable level of intermodule coupling exhibited by the TSF;

d) a description of the minimum acceptable level of cohesion exhibited the modules of the TSF.

Complexity can for example be measured in the number of decision points and logical paths of execution that code takes. Software engineering literature cites complexity as a negative characteristic of software because it impedes understanding of the logic and flow of the code. Another impediment to the understanding of code is the presence of code that is unnecessary, in that it is unused or redundant.

Design complexity minimisation is a key characteristic of a reference validation mechanism, the purpose of which is to arrive at a TSF that is easily understood so that it can be completely analysed.

See also CC Part 3, A.3 for additional information on TSF internals.

The consideration in that annex and those made in the preceding paragraphs of this work unit are mainly derived from common knowledge about procedural software. For other types of technologies used in the TOE - such as non-procedural software (e.g. object-oriented programming), widespread commodity hardware (e.g. PC microprocessors), and special-purpose hardware (e.g. smart-card processors) - the evaluation authority should be consulted for determining the adequacy of criteria for being "well-structured" and "not overly complex".

The evaluator is reminded to be open for plausible definitions given by the developer. If. for example, a smart card developer can justify that the metrics used to measure complexity are an industry standard in their field, this should usually be sufficient for acceptance of such metrics.

CC Part 3 ADV_INT.3.2C: *The TSF internals description shall demonstrate that the entire TSF is well-structured and is not overly complex.*

13.6.3.4.3 Work unit ADV_INT.3-2

The evaluator ***shall examine*** the TSF internals description to determine that it demonstrates that the TSF is well-structured and not overly complex.

The evaluator examines the internals description to ensure that it provides a sound explanation of how the TSF meets the criteria from ADV_INT.3-1

For example, it would explain how the procedural software portions of the TSF meet the following:

a) that there is a one-to-one correspondence between the modules identified in the TSF and the modules described in the TOE design (ADV_TDS);

b) how the TSF design is a reflection of the modular decomposition process;

c) a justification for all instances where the coding standards were not used or met;

d) a justification for any coupling or cohesion outside the acceptable bounds;

e) how the modular decomposition process reduces complexity.

13.6.3.5 Action ADV_INT.3.2E

13.6.3.5.1 Work unit ADV_INT.3-3

The evaluator ***shall determine*** that the entire TOE design is well-structured and not overly complex.

The evaluator examines the TOE design description of the TSF to verify the accuracy of the justification. For example, a sample of the TOE design is analysed to determine its adherence to the design standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope

The description of the TOE's decomposition into subsystems and modules will make the argument that the TSF is well-structured self-evident. Verification that the procedures for structuring the TSF (as examined in ALC_TAT) are being followed will make it self-evident that the TSF is well-structured.

Using the metrics defined by the developer for measuring the complexity of the design will show if the metrics is met. If the metrics is only defined for the implementation representation and not for the TOE design (note that adequateness of the metrics was considered already in work unit ADV_INT.3-1), there may be no need for using the metrics in this work unit, the complexity-issue is then covered by the next work unit.

13.6.3.5.2 Work unit ADV_INT.3-4

The evaluator ***shall determine*** that the entire TSF is well-structured and not overly complex.

If ADV_IMP is not part of the claimed assurance, then this work unit is not applicable and is therefore considered to be satisfied.

The evaluator examines a sample of the TSF to verify the accuracy of the internals description. For example, a sample of the procedural software portions of the TSF is analysed to determine its cohesion and coupling, its adherence to the coding standards, etc. As with all areas where the evaluator performs activities on a subset the evaluator provides a justification of the sample size and scope.

Similarly, the evaluator applies the metric for complexity as defined by the developer and examined in work unit ADV_INT.3-1 to either a sample of the implementation representation or the complete implementation representation (this may depend on the metric) and verifies that the metric is in fact met. The evaluator may only restrict their application of the metrics to a sample if the developer has provided the results of the application of the metrics for the entire TSF and the sampling serves as means to convince the evaluator that the application as done by the developer was correct (similar to the evaluator's sampling of functional testing already done by the developer).

13.7 Formal TSF model (ADV_SPM)

13.7.1 Evaluation of sub-activity (ADV_SPM.1)

13.7.1.1 Objectives

The objective of this sub-activity is to determine whether the formal model and its formal properties clearly and consistently represent the TSF and the TOE, as defined by the SFRs and the security objectives for the TOE of the ST.

13.7.1.2 Inputs

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the formal model for the TSF and the supporting explanatory text (CC Part 3 ADV_SPM.1.1D);

d) the set of formal properties for the TOE and the supporting explanatory text (CC Part 3 ADV_SPM.1.2D);

e) the formal proof that the model satisfies the formal properties and the supporting explanatory text (CC Part 3 ADV_SPM.1.3D);

f) a correspondence rationale between the formal model and the functional specification (CC Part 3 ADV_SPM.1.4D);

g) a semiformal demonstration of correspondence between the formal model and any semiformal functional specification (CC Part 3 ADV_SPM.1.5D);

h) a formal proof of correspondence between the formal model and any formal functional specification (CC Part 3 ADV_SPM.1.6D);

i) all the tools used for the formal model, the formal properties, proofs and demonstrations (CC Part 3 ADV_SPM.1.7D).

13.7.1.3 Application notes

This activity applies to cases where the developer has provided a formal Security Policy Model (SPM) of the TSF.

A formal security model is a formal representation of the essential aspects of security, i.e. the TSF and their relationship to the behaviour of the TOE. More specifically, the formal model is a formal representation of the TSF as defined by the complete set of SFRs described in the ST. The set of formal properties covers all the security objectives for the TOE.

The creation of a formal Security Policy Model (SPM) of the TSF must be complete with respect to the ST; such a model helps to identify and eliminate ambiguous, inconsistent, contradictory, or unenforceable elements and to avoid any misunderstanding related to the scope. To this end, the evaluation must determine whether the formal model and the formal properties completely cover the ST and accept only STs and SPMs that match in scope. Once the TOE has been built, the formal model serves the evaluation effort by contributing to the evaluator's judgement of how well the developer has understood the TSF being implemented and whether there are inconsistencies between the formal properties as defined by the security objectives for the TOE of the ST and the TOE itself. The confidence gained by formally proving properties of the model is accompanied by confidence gained by defining a correspondence rationale between the formal model and the TSF functional specification (as defined for ADV_FSP). The correspondence rationale consists of a formal proof when mapping to formal aspects of the TSF functional specification and semiformal demonstration otherwise. A combination of different formal systems (modelling languages, tools, proof systems) can be used for different parts of the ST (SFRs & Security Objectives) and correspondence rationales.

13.7.1.4 Action ADV_SPM.1.1E

13.7.1.4.1 General

CC Part 3 ADV_SPM.1.1C: *The formal model, properties and proofs shall be defined using a well-founded mathematical theory.*

13.7.1.4.2 Work unit ADV_SPM.1-1

The evaluator ***shall examine*** the formal model, the formal properties and the formal proofs to determine that their expression relies on a well-founded syntax and semantics.

This work unit consists of verifying and ensuring the well-foundedness of the underlying mathematical concepts used for expressing the formal model and for proving its formal properties and formal correspondence to the FSP (if applicable).

The evaluator identifies the mathematical concepts upon which the formal model, properties and proofs are based and gathers relevant scientific bibliography in the context of the evaluation, e.g. on the basis of the provided description of used tools and additional references. This literature research should facilitate the identification of “pitfalls” that would allow the introduction of paradoxes and/or fallacies.

The evaluator confirms that the underlying mathematical concepts are well-founded.

This work unit applies to all the different mathematical theories that are used to provide the formal model, properties and proofs.

CC Part 3 ADV_SPM.1.2C: *The explanatory text shall cover the entire formal model, properties and proofs, including instructions for reproducing the proofs and the correspondence rationale, and it shall provide a rationale for the modelling and verification choices.*

13.7.1.4.3 Work unit ADV_SPM.1-2

The evaluator ***shall examine*** the explanatory text to determine that it explains all parts of the formal model, properties and proofs including the unproven assertions, the chosen verification strategy and the instructions needed to reproduce the proofs and the correspondence rationale.

This work unit consists in examining the explanatory text to determine that it describes all parts of the formal model for all parts of the TSF, the formal properties for all the security objectives for the TOE and the formal proofs, and that it justifies the adopted modelling and verification strategy.

The explanatory text consists of supporting descriptions and clarifications, for example, the formal notation and its usage, or the formal/mathematical representation of the concepts. The explanatory text covers all the unproven assertions on which the formal model or the formal proof rely, and identifies all the formal model and formal proof elements which correspond to the security objectives concerning the TOE environment.

The explanatory text is expected to provide sufficient details to enable the identification of all parts of the formal model and all the formal properties as well as the correct understanding of the formal model, the properties, the proofs and the reasoning behind the modelling and verification choices.

The explanatory text is expected to support the correspondence between formal and natural language concepts and properties as stated in the ST and functional specification.

Note: Comments included directly in the source code of the formal model or proofs can help but they do not exclude provision of the explanatory text itself and are instead complementary to the latter.

CC Part 3 ADV_SPM.1.3C: *The formal model shall cover the complete set of SFRs that define the TSF.*

13.7.1.4.4 Work unit ADV_SPM.1-3

The evaluator ***shall examine*** the formal model, the definition of the SFRs, and the explanatory text, to determine that the formal model covers the complete set of SFRs defined in the ST.

In determining completeness of the formal model, the evaluator maps the elements of the SFRs to the formal model's elements. Omission of any SFR from the formal modelling leads to a fail verdict being assigned to this work unit.

13.7.1.4.5 Work unit ADV_SPM.1-4

The evaluator ***shall examine*** the formal model, the definition of the SFRs and the explanatory text to determine that the formal model provides an accurate representation of the SFRs.

This work unit consists in determining whether the formal model provides an accurate representation of the SFRs.

The evaluator examines the SFRs, the formal model, and the explanatory text to establish the accuracy of the correspondence between the formal concepts and the definition of the SFRs.

In determining accuracy, the evaluator verifies that the definition of the SFRs is reflected within the formal model and that the explanatory text describing each formal element in the model accurately addresses the concepts and the intent of the corresponding SFR. The evaluator also ensures that the formal model defines a representation of the TSF that is appropriate for the statement of the formal properties to be proven.

NOTE The formal model aims to cover the entire set of SFRs, which goes beyond access and flow control policies stated by means of FDP_ACC and FDP_IFC components. All the SFRs defined in the ST are concerned, including, for example, non-repudiation, privacy or authentication SFRs.

The formal model and the definition of the SFRs may use different abstractions. This does not affect the accuracy of the formal model if the formal properties are not affected.

EXAMPLE 1 A TSF that is formally modelled on the basis of state transitions would include a definition of the states, identify the initial state(s), describe the necessary conditions to move from one state to the next, and characterize the secure state(s).

EXAMPLE 2 In the presence of formal properties related to authentication, if SFRs state that access control is necessary to the granularity of a single individual, then a formal model describing the TSF behaviour in the context of controlling groups of users would not be accurate.

CC Part 3 ADV_SPM.1.4C: *The formal properties shall cover the complete set of security objectives for the TOE.*

13.7.1.4.6 Work unit ADV_SPM.1-5

The evaluator ***shall examine*** the formal properties, the security objectives for the TOE and the explanatory text to determine that the formal properties cover the complete set of security objectives for the TOE defined in the ST.

In determining completeness of the formal properties, the evaluator maps the elements of the security objectives for the TOE to the elements of the formal properties' statements. Omission of any security objective from the formally proven properties of the TOE always leads to a fail verdict being assigned to this work unit.

13.7.1.4.7 Work unit ADV_SPM.1-6

The evaluator ***shall examine*** the definition of the formal properties to determine that they accurately represent the security objectives for the TOE as given in the ST.

This work unit consists in determining whether the formal properties provide an accurate representation of the security objectives for the TOE.

The evaluator examines the definition of the security objectives for the TOE, the formal model, the formal properties, and the explanatory text to establish the accuracy of the correspondence between the formal and the informal concepts.

CC Part 3 ADV_SPM.1.5C: *The formal proof shall show that the formal model satisfies all the formal properties and that the consistency of the underlying mathematical theory is preserved.*

13.7.1.4.8 Work unit ADV_SPM.1-7

The evaluator ***shall reproduce*** the formal proof by following the instructions provided by the developer as part of the explanatory text and confirm that the obtained results are identical.

This work unit consists in determining that mathematical proof of the formal properties exists and is reproducible, that is, in verifying the formal proof itself, using the instructions and tools provided by the developer.

The evaluator checks whether the explanatory text for the formal proof contains step-by-step instructions that are applicable and lead to results that are identical to those provided by the developer.

The evaluator reproduces the formal proof using the formal model, the tools and the instructions supplied by the developer, and manually checks all the formal proofs provided on paper.

13.7.1.4.9 Work unit ADV_SPM.1-8

The evaluator ***shall examine*** the formal proof to determine that the formal model satisfies all the formal properties and that the consistency of the underlying mathematical theory is preserved.

This work unit consists in determining that mathematical proof exists that the formal model satisfies the entire set of formal properties and that a state which does not preserve the formal properties cannot be reached.

The evaluator relies on the formal proof, the explanatory text and on the principles of the underlying formal theory and its implementation to determine that the consistency of the underlying mathematical theory is preserved.

The evaluator checks whether the explanatory text contains sufficient details for understanding the formal proof and all the unproven assertions that are used to conduct the proof.

The evaluator determines that none of the unproven assertions used in the proof invalidate the well-foundedness of the underlying formal theory. By doing this, the evaluator determines that the formal model enforces the formal properties and that the arguments used in the formal proof are valid and do not lead to contradictions.

CC Part 3 ADV_SPM.1.6C: *The correspondence rationale shall show that the formal properties proven for the formal model hold for the functional specification.*

13.7.1.4.10 Work unit ADV_SPM.1-9

The evaluator ***shall examine*** the correspondence rationale to determine that all the formal properties proven to be satisfied by the formal model hold for the functional specification.

This work unit consists in verifying that the functional specification reflects all the concepts and formal properties which are modelled and proven to hold in the formal model.

The correspondence rationale provided by the developer is expected to provide tables, diagrams and/or explanatory text to describe the relationship between the entire functional specification and the formal model.

The evaluator examines the correspondence rationale to determine whether there is sufficient evidence to conclude that the formal properties satisfied by the formal model hold in the functional specification. Establishing whether the abstraction used for modelling the TSF is relevant is an essential part of evaluating the rationale behind the preservation of the model's formal properties by the functional specification. Therefore, the evaluator shall examine the correspondence rationale to determine whether the abstraction level used for the formal model and properties is appropriate.

This is particularly true in the context of ADV_FSP.4 in which case the evaluator must make a judgement regarding the abstraction relevance of the formal model without the support of formal proofs.

CC Part 3 ADV_SPM.1.7C: *The semiformal demonstration of correspondence shall show that the formal properties proven for the formal model hold for any semiformal functional specification.*

13.7.1.4.11 Work unit ADV_SPM.1-10

If the functional specification does not contain a semiformal description, this work unit is not applicable and is therefore considered to be satisfied. Otherwise, the evaluator ***shall check*** that

the semiformal demonstration of correspondence shows that the formal properties proven for the formal model hold in the semiformal functional specification.

A semiformal demonstration of correspondence is one that results from a structured approach with a substantial degree of rigor (in terms of completeness and correctness), but is not as rigorous as a mathematical proof. Such a semiformal correspondence limits the subjective interpretations of its terms, and so it provides less ambiguity than would exist in an informal correspondence.

The evaluator examines the semiformal proof of correspondence to determine whether there is sufficient evidence to conclude that the formal properties satisfied by the formal model hold in the semi-formal functional specification. To this end, the evaluator uses the tools provided by the developer.

CC Part 3 ADV_SPM.1.8C: *The formal proof of correspondence shall show that the formal properties proven for the formal model hold for any formal functional specification.*

13.7.1.4.12 Work unit ADV_SPM.1-11

If the functional specification does not contain a formal description, this work unit is not applicable and is therefore considered to be satisfied.

Otherwise, the evaluator ***shall verify*** that the formal proof of correspondence shows that the formal properties proven for the formal model hold in the formal functional specification.

A formal proof of correspondence removes all subjective interpretations of its terms by enlisting well-established mathematical concepts to define the syntax and semantics of the formal notation and the proof rules that support logical reasoning.

The evaluator examines the formal proof of correspondence to determine whether there is sufficient evidence to conclude that the formal properties satisfied by the formal model hold in the formal functional specification. To this end, the evaluator reproduces the formal correspondence proof by using the tools provided by the developer. For a paper proof, the evaluator verifies the proof manually.

CC Part 3 ADV_SPM.1.9C: *Any tool used to model or to prove the formal properties or the relationship between the formal model and the functional specification shall be well-defined and unambiguously identified and it shall be accompanied by documentation and a rationale of the tool's suitability and trustworthiness.*

13.7.1.4.13 Work unit ADV_SPM.1-12

The evaluator ***shall examine*** whether an unambiguous identification of the tools used for formal modelling and proofs and semiformal demonstrations (if applicable) is provided by the developer and determine whether the presented results can be reproduced with the identified tools.

The documentation of the tools is included in the evaluator's examination.

The evaluator verifies that the tools versions and the environment in which they were used are unambiguously identified by the developer.

The evaluator checks that the identified tools versions are not subject to known soundness problems.

The evaluator checks whether the results obtained with the identified tools are identical to those provided by the developer.

13.7.1.4.14 Work unit ADV_SPM.1-13

The evaluator shall determine the consistency of the analysis of the suitability and trustworthiness of the formal modelling, proof and demonstration tools. The evaluator shall ensure that the tools have a sound mathematical foundation.

This work unit consists in examining the consistency of the rationale provided for the suitability and trustworthiness of the used tools.

The evaluator determines whether the formal theory is consistent with its implementation in the tools. For example, it can be the case that there are slight deviations in the interpretations of certain syntax elements between the mathematical definition and the implementation in the tools.

The evaluator examines the consistency of all the modelling and verification tools used. For example, the developer can use a modelling tool A for state-charts to express the semantics of the system and an interactive theorem prover B to verify its formal properties. In between, there is a translation from the output format of A to the representation used by/input format of B. In addition to examining the correctness of this twofold approach, the soundness of the mapping and translation from the output format of A to the representation used by/input format of B must be examined too.

One can manually insert modelling errors to show that the formalism is indeed capable of detecting flaws. However, the capability to detect such flaws is necessary but not sufficient.

13.7.1.4.15 Work unit ADV_SPM.1-14

The evaluator ***shall examine*** the documentation of the tools used for formal modelling and proofs and for semiformal demonstrations (if applicable) to determine that it unambiguously defines the meaning of all statements, all model- and proof-dependent options, as well as all conventions and directives used.

The tools documentation should cover all statements used in the formal models, proofs and demonstrations, and for each such statement should provide a clear and unambiguous definition of the purpose and effect of that statement. This work should be performed in parallel with the evaluator's examination of the semiformal and formal models and specifications. The critical test is whether the evaluator can understand the source code of the semiformal or formal models, proofs and demonstrations (if applicable).

For any identified tool, the evaluator checks whether:

a) any terms, abbreviations and acronyms that are used in a context other than that accepted by normal usage are defined;

b) the notation used provides rules to determine the meaning of syntactical valid constructs.

For any identified tool used for formal modelling and proofs, the evaluator determines whether:

a) its semantics is well-defined;

b) it uses a formal syntax that provides rules to unambiguously recognise constructs;

c) it supports logical reasoning using well-established mathematical concepts;

d) it helps to prevent misuse and the introduction of contradictions by issuing warnings or hints, rejecting invalid constructions or providing specific instructions within the usage manual.

13.8 TOE design (ADV_TDS)

13.8.1 Evaluation of sub-activity (ADV_TDS.1)

13.8.1.1 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) security architecture description;

d) the TOE design.

13.8.1.2 Action ADV_TDS.1.1E

13.8.1.2.1 General

CC Part 3 ADV_TDS.1.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

13.8.1.2.2 Work unit ADV_TDS.1-1

The evaluator ***shall examine*** the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.

The evaluator ensures that all of the subsystems of the TOE are identified. This description of the TOE will be used as input to work unit ADV_TDS.1-2, where the parts of the TOE that make up the TSF are identified. That is, this requirement is on the entire TOE rather than on only the TSF.

The TOE (and TSF) may be described in multiple layers of abstraction (i.e. subsystems and modules). Depending upon the complexity of the TOE, its design may be described in terms of subsystems and modules, as described in CC Part 3, Annex A, ADV_TDS: Subsystems and Modules. At this level of assurance, the decomposition only need be at the "subsystem" level.

In performing this activity, the evaluator examines other evidence presented for the TOE (e.g., ST, operator user guidance) to determine that the description of the TOE in such evidence is consistent with the description contained in the TOE design.

CC Part 3 ADV_TDS.1.2C: *The design shall identify all subsystems of the TSF.*

13.8.1.2.3 Work unit ADV_TDS.1-2

The evaluator ***shall examine*** the TOE design to determine that all subsystems of the TSF are identified.

In work unit ADV_TDS.1-1 all of the subsystems of the TOE were identified, and a determination made that the non-TSF subsystems were correctly characterised. Building on that work, the subsystems that were not characterised as non-TSF subsystems should be precisely identified. The evaluator determines that, of the hardware and software installed and configured according to the Preparative procedures (AGD_PRE) guidance, each subsystem has been accounted for as either one that is part of the TSF, or one that is not.

If TSFs are defined in terms of sub-TSFs for multi assurance the evaluator ***shall examine*** that the combination of all sub-TSF is consistent and does not omit relevant information for each sub-TSF considering the relevant decomposition level.

CC Part 3 ADV_TDS.1.3C: *The design shall provide the behaviour summary of each SFR-supporting or SFR-non-interfering TSF subsystem.*

13.8.1.2.4 Work unit ADV_TDS.1-3

The evaluator ***shall examine*** the TOE design to determine that each SFR-supporting or SFR-non-interfering subsystem of the TSF is described such that the evaluator can determine that the subsystem is SFR-supporting or SFR-non-interfering.

SFR-supporting and SFR-non-interfering subsystems do not need to be described in detail as to how they function in the system. However, the evaluator makes a determination, based on the evidence provided by the developer, that the subsystems that do not have high-level descriptions are SFR-supporting or SFR-non-interfering. Note that if the developer provides a uniform level of detailed documentation then this work unit will be largely satisfied, since the point of categorising the subsystems is to allow the developer to provide less information for SFR-supporting and SFR-non-interfering subsystems than for SFR-enforcing subsystems.

An SFR-supporting subsystem is one that is depended on by an SFR-enforcing subsystem in order to implement an SFR, but does not play as direct a role as an SFR-enforcing subsystem. An SFR-non-interfering subsystem is one that is not depended upon, in either a supporting or enforcing role, to implement an SFR.

CC Part 3 ADV_TDS.1.4C: *The design shall summarize the SFR-enforcing behaviour of the SFR-enforcing subsystems.*

13.8.1.2.5 Work unit ADV_TDS.1-4

The evaluator ***shall examine*** the TOE design to determine that it provides a complete, accurate, and high-level summary of the SFR-enforcing behaviour of the SFR-enforcing subsystems.

The developer may designate subsystems as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the subsystems have been categorised by the developer or not, it is the evaluator's responsibility to determine that the subsystems have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non- interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular subsystem.

SFR-enforcing behaviour refers to how a subsystem provides the functionality that implements an SFR. The goal of evaluator's assessment is to give the evaluator with an understanding of the way each SFR-enforcing subsystem works. The information provided for the behaviour summary does not have to be as detailed as that provided by the behaviour description. For example, data structures or data items will likely not need to be described in detail. It is the evaluator's determination, however, with respect to what "high-level" means for a particular TOE, and the evaluator obtains enough information from the developer (even if it turns out to be equivalent to information provided for subsystem behaviour) to make a sound verdict for this work unit.

The evaluator is cautioned, however, that "perfect" assurance is not a goal nor required by this work unit, so judgement will have to be exercised in determine the amount and composition of the evidence required to make a verdict on this work unit.

To determine completeness and accuracy, the evaluator examines other information available (e.g., functional specification, security architecture description). Summaries of functionality in these documents should be consistent with what is provided for evidence for this work unit.

CC Part 3 ADV_TDS.1.5C: *The design shall provide a description of the interactions among SFR-enforcing subsystems of the TSF, and between the SFR-enforcing subsystems of the TSF and other subsystems of the TSF.*

13.8.1.2.6 Work unit ADV_TDS.1-5

The evaluator ***shall examine*** the TOE design to determine that interactions between the subsystems of the TSF are described.

The goal of describing the interactions between the SFR-enforcing subsystems and other subsystems is to help provide the reader a better understanding of how the TSF performs it functions. These interactions do not need to be characterised at the implementation level (e.g., parameters passed from one routine in a subsystem to a routine in a different subsystem; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular subsystem that are going to be used by another subsystem need to be covered in this discussion. Any control relationships between subsystems (e.g., a subsystem responsible for configuring a rule base for a firewall system and the subsystem that actually implements these rules) should also be described.

The evaluators need to use their own judgement in assessing the completeness of the description. If the reason for an interaction is unclear, or if there are SFR-related interactions (discovered, for instance, in examining the descriptions of subsystem behaviour) that do not appear to be described, the evaluator ensures that this information is provided by the developer. However, if the evaluator can determine that interactions among a particular set of subsystems, while incompletely described by the developer, will not aid in understanding the overall functionality nor security functionality provided by the TSF, then the evaluator may choose to consider the description sufficient, and not pursue completeness for its own sake.

CC Part 3 ADV_TDS.1.6C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

13.8.1.2.7 Work unit ADV_TDS.1-6

The evaluator ***shall examine*** the TOE design to determine that it contains a complete and accurate mapping from the TSFI described in the functional specification to the subsystems of the TSF described in the TOE design.

The subsystems described in the TOE design provide a description of how the TSF works at a detailed level for SFR-enforcing portions of the TSF, and at a higher level for other portions of the TSF. The TSFI provide a description of how the implementation is exercised. The evidence from the developer identifies the subsystem that is initially involved when an operation is requested at the TSFI, and identify the various subsystems that are primarily responsible for implementing the functionality. Note that a complete "call tree" for each TSFI is not required for this work unit.

The evaluator assesses the completeness of the mapping by ensuring that all of the TSFI map to at least one subsystem. The verification of accuracy is more complex.

The first aspect of accuracy is that each TSFI is mapped to a subsystem at the TSF boundary. This determination can be made by reviewing the subsystem description and interactions, and from this information determining its place in the architecture. The next aspect of accuracy is that the mapping makes sense. For instance, mapping a TSFI dealing with access control to a subsystem that checks passwords is not accurate. The evaluator should again use judgement in making this determination. The goal is that this information aids the evaluator in understanding the system and implementation of the SFRs, and ways in which entities at the TSF boundary can interact with the TSF. The bulk of the assessment of whether the SFRs are described accurately by the subsystems is performed in other work units.

13.8.1.3 Action ADV_TDS.1.2E

13.8.1.3.1 Work unit ADV_TDS.1-7

The evaluator ***shall examine*** the TOE security functional requirements and the TOE design, to determine that all ST security functional requirements are covered by the TOE design.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 Subset access control component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 Subset access control assignment, and these ten rules were implemented in specific places within fifteen modules, it would be inadequate for the evaluator to map FDP_ACC.1 Subset access control to one subsystem and claim the work unit had been completed. Instead, the evaluator would map FDP_ACC.1 Subset access control (rule 1) to subsystem A, behaviours x, y, and z; FDP_ACC.1 Subset access control (rule 2) to subsystem A, behaviours x, p, and q; etc.

13.8.1.3.2 Work unit ADV_TDS.1-8

The evaluator ***shall examine*** the TOE design to determine that it is an accurate instantiation of all security functional requirements.

The evaluator ensures that each security requirement listed in the TOE security functional requirements subclause of the ST has a corresponding design description in the TOE design that accurately details how the TSF meets that requirement. This requires that the evaluator identify a collection of subsystems that are responsible for implementing a given functional requirement, and then examine those subsystems to understand how the requirement is implemented. Finally, the evaluator would assess whether the requirement was accurately implemented.

As an example, if the ST requirements specified a role-based access control mechanism, the evaluator would first identify the subsystems that contribute to this mechanism's implementation. This can be done by in-depth knowledge or understanding of the TOE design or by work done in the previous work unit. Note that this trace is only to identify the subsystems, and is not the complete analysis.

The next step would be to understand what mechanism the subsystems implemented. For instance, if the design described an implementation of access control based on UNIX-style protection bits, the design would not be an accurate instantiation of those access control requirements present in the ST example used above. If the evaluator cannot determine that the mechanism was accurately implemented because of a lack of detail, the evaluator would have to assess whether all of the SFR-enforcing subsystems have been identified, or if adequate detail had been provided for those subsystems.

13.8.2 Evaluation of sub-activity (ADV_TDS.2)

13.8.2.1 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) security architecture description;

d) the TOE design.

13.8.2.2 Action ADV_TDS.2.1E

13.8.2.2.1 General

CC Part 3 ADV_TDS.2.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

13.8.2.2.2 Work unit ADV_TDS.2-1

The evaluator ***shall examine*** the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.

The evaluator ensures that all of the subsystems of the TOE are identified. This description of the TOE will be used as input to work unit ADV_TDS.2-2, where the parts of the TOE that make up the TSF are identified. That is, this requirement is on the entire TOE rather than on only the TSF.

The TOE (and TSF) may be described in multiple layers of abstraction (i.e. subsystems and modules). Depending upon the complexity of the TOE, its design may be described in terms of subsystems and modules, as described in CC Part 3, Annex A.4, ADV_TDS: Subsystems and Modules. At this level of assurance, the decomposition only need be at the "subsystem" level.

In performing this activity, the evaluator examines other evidence presented for the TOE (e.g., ST, operator user guidance) to determine that the description of the TOE in such evidence is consistent with the description contained in the TOE design.

CC Part 3 ADV_TDS.2.2C: *The design shall identify all subsystems of the TSF.*

13.8.2.2.3 Work unit ADV_TDS.2-2

The evaluator ***shall examine*** the TOE design to determine that all subsystems of the TSF are identified.

In work unit ADV_TDS.2-1 all of the subsystems of the TOE were identified, and a determination made that the non-TSF subsystems were correctly characterised. Building on that work, the subsystems that were not characterised as non-TSF subsystems should be precisely identified. The evaluator determines that, of the hardware and software installed and configured according to the Preparative procedures (AGD_PRE) guidance, each subsystem has been accounted for as either one that is part of the TSF, or one that is not.

CC Part 3 ADV_TDS.2.3C: *The design shall provide the behaviour summary of each SFR non-interfering subsystem of the TSF.*

13.8.2.2.4 Work unit ADV_TDS.2-3

The evaluator ***shall examine*** the TOE design to determine that each SFR-non-interfering subsystem of the TSF is described such that the evaluator can determine that the subsystem is SFR-non-interfering.

SFR-non-interfering subsystems do not need to be described in detail as to how they function in the system. However, the evaluator makes a determination, based on the evidence provided by the developer, that the subsystems that do not have detailed descriptions are SFR-non-interfering. Note that if the developer provides a uniform level of detailed documentation then this work unit will be largely satisfied, since the point of categorising the subsystems is to allow the developer to provide less information for SFR-non-interfering subsystems than for SFR-enforcing and SFR-supporting subsystems.

An SFR-non-interfering subsystem is one on which the SFR-enforcing and SFR-supporting subsystems have no dependence; that is, they play no role in implementing SFR functionality.

CC Part 3 ADV_TDS.2.4C: *The design shall describe the SFR-enforcing behaviour of the SFR-enforcing subsystems.*

13.8.2.2.5 Work unit ADV_TDS.2-4

The evaluator ***shall examine*** the TOE design to determine that it provides a complete, accurate, and detailed description of the SFR-enforcing behaviour of the SFR-enforcing subsystems.

The developer may designate subsystems as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the subsystems have been categorised by the developer or not, it is the evaluator's responsibility to determine that the subsystems have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular subsystem.

SFR-enforcing behaviour refers to *how* a subsystem provides the functionality that implements an SFR. While not at the level of an algorithmic description, a detailed description of behaviour typically discusses how the functionality is provided in terms of what key data and data structures are, what control relationships exist within a subsystem, and how these elements work together to provide the SFR-enforcing behaviour. Such a description also references SFR-supporting behaviour, which the evaluator should consider in performing subsequent work units.

To determine completeness and accuracy, the evaluator examines other information available (e.g., functional specification, security architecture description). Descriptions of functionality in these documents should be consistent with what is provided for evidence for this work unit.

CC Part 3 ADV_TDS.2.5C: *The design shall summarize the SFR-supporting and SFR-non-interfering behaviour of the SFR-enforcing subsystems.*

13.8.2.2.6 Work unit ADV_TDS.2-5

The evaluator ***shall examine*** the TOE design to determine that it provides a complete and accurate high-level summary of the SFR-supporting and SFR-non-interfering behaviour of the SFR-enforcing subsystems.

The developer may designate subsystems as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the subsystems have been categorised by the developer or not, it is the evaluator's responsibility to determine that the subsystems have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular subsystem.

In contrast to the previous work unit, this work unit calls for the evaluator to assess the information provided for SFR-enforcing subsystems that is SFR-supporting or SFR-non-interfering. The goal of this assessment is two-fold. First, it should provide the evaluator greater understanding of the way each subsystem works. Second, this assessment will help the evaluator to determine that all SFR-enforcing behaviour exhibited by a SFR-enforcing subsystem has been described. Unlike the previous work unit, the information provided for the SFR-supporting or

SFR-non-interfering behaviour does not have to be as detailed as that provided by the SFR-enforcing behaviour. For example, data structures or data items that do not pertain to SFR-enforcing functionality will likely not need to be described in detail, if at all. It is the evaluator's determination, however, with respect to what "high-level" means for a particular TOE, and the evaluator obtains enough information from the developer (even if it turns out to be equivalent to information provided for the parts of the subsystem that are SFR-enforcing) to make a sound verdict for this work unit.

The evaluator is cautioned, however, that "perfect" assurance is not a goal nor required by this work unit, so judgement will have to be exercised in determine the amount and composition of the evidence required to make a verdict on this work unit.

To determine completeness and accuracy, the evaluator examines other information available (e.g., functional specification, security architecture description). Summaries of functionality in these documents should be consistent with what is provided for evidence for this work unit. In particular, the functional specification should be used to determine that the behaviour required to implement the TSF Interfaces described by the functional specification are completely described by the subsystem, since the behaviour will either be SFR-enforcing, SFR-supporting or SFR-non-interfering.

CC Part 3 ADV_TDS.2.6C: *The design shall summarize the behaviour of the SFR-supporting subsystems.*

13.8.2.2.7 Work unit ADV_TDS.2-6

The evaluator ***shall examine*** the TOE design to determine that it provides a complete and accurate high-level summary of the behaviour of the SFR-supporting subsystems.

The developer may designate subsystems as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the subsystems have been categorised by the developer or not, it is the evaluator's responsibility to determine that the subsystems have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular subsystem.

In contrast to the previous two work units, this work unit calls for the developer to provide (and the evaluator to assess) information about SFR supporting subsystems. Such subsystems should be referenced by the descriptions of the SFR-enforcing subsystems, as well as by the descriptions of interactions in work unit ADV_TDS.2-7. The goal of evaluator's assessment, like that for the previous work unit, is two-fold. First, it should provide the evaluator with an understanding of the way each SFR-supporting subsystem works. Second, the evaluator determines that the behaviour is summarized in enough detail so that the way in which the subsystem supports the SFR-enforcing behaviour is clear, and that the behaviour is not itself SFR-enforcing. The information provided for SFR-supporting subsystem's behaviour does not have to be as detailed as that provided by the SFR-enforcing behaviour. For example, data structures or data items that do not pertain to SFR-enforcing functionality will likely not need to be described in detail, if at all. It is the evaluator's determination, however, with respect to what "high-level" means for a particular TOE, and the evaluator obtains enough information from the developer (even if it turns out to be equivalent to information provided for the parts of the subsystem that are SFR-enforcing) to make a sound verdict for this work unit.

The evaluator is cautioned, however, that "perfect" assurance is not a goal nor required by this work unit, so judgement will have to be exercised in determine the amount and composition of the evidence required to make a verdict on this work unit.

To determine completeness and accuracy, the evaluator examines other information available (e.g., functional specification, security architecture description). Summaries of functionality in these documents should be consistent with what is provided for evidence for this work unit.

CC Part 3 ADV_TDS.2.7C: *The design shall provide a description of the interactions among all subsystems of the TSF.*

13.8.2.2.8 Work unit ADV_TDS.2-7

The evaluator ***shall examine*** the TOE design to determine that interactions between the subsystems of the TSF are described.

The goal of describing the interactions between the subsystems is to help provide the reader a better understanding of how the TSF performs it functions. These interactions do not need to be characterised at the implementation level (e.g., parameters passed from one routine in a subsystem to a routine in a different subsystem; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular subsystem that are going to be used by another subsystem need to be covered in this discussion. Any control relationships between subsystems (e.g., a subsystem responsible for configuring a rule base for a firewall system and the subsystem that actually implements these rules) should also be described.

It should be noted while the developer should characterise all interactions between subsystems, the evaluators need to use their own judgement in assessing the completeness of the description. If the reason for an interaction is unclear, or if there are SFR-related interactions (discovered, for instance, in examining the descriptions of subsystem behaviour) that do not appear to be described, the evaluator ensures that this information is provided by the developer. However, if the evaluator can determine that interactions among a particular set of subsystems, while incompletely described by the developer, will not aid in understanding the overall functionality nor security functionality provided by the TSF, then the evaluator may choose to consider the description sufficient, and not pursue completeness for its own sake.

CC Part 3 ADV_TDS.2.8C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

13.8.2.2.9 Work unit ADV_TDS.2-8

The evaluator ***shall examine*** the TOE design to determine that it contains a complete and accurate mapping from the TSFI described in the functional specification to the subsystems of the TSF described in the TOE design.

The subsystems described in the TOE design provide a description of how the TSF works at a detailed level for SFR-enforcing portions of the TSF, and at a higher level for other portions of the TSF. The TSFI provide a description of how the implementation is exercised. The evidence from the developer identifies the subsystem that is initially involved when an operation is requested at the TSFI, and identify the various subsystems that are primarily responsible for implementing the functionality. Note that a complete "call tree" for each TSFI is not required for this work unit.

The evaluator assesses the completeness of the mapping by ensuring that all of the TSFI map to at least one subsystem. The verification of accuracy is more complex.

The first aspect of accuracy is that each TSFI is mapped to a subsystem at the TSF boundary. This determination can be made by reviewing the subsystem description and interactions, and from

this information determining its place in the architecture. The next aspect of accuracy is that the mapping makes sense. For instance, mapping a TSFI dealing with access control to a subsystem that checks passwords is not accurate. The evaluator should again use judgement in making this determination. The goal is that this information aids the evaluator in understanding the system and implementation of the SFRs, and ways in which entities at the TSF boundary can interact with the TSF. The bulk of the assessment of whether the SFRs are described accurately by the subsystems is performed in other work units.

13.8.2.3 Action ADV_TDS.2.2E

13.8.2.3.1 Work unit ADV_TDS.2-9

The evaluator ***shall examine*** the TOE security functional requirements and the TOE design, to determine that all ST security functional requirements are covered by the TOE design.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 Subset access control component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 Subset access control assignment, and these ten rules were implemented in specific places within fifteen modules, it would be inadequate for the evaluator to map FDP_ACC.1 Subset access control to one subsystem and claim the work unit had been completed. Instead, the evaluator would map FDP_ACC.1 Subset access control (rule 1) to subsystem A, behaviours x, y, and z; FDP_ACC.1 Subset access control (rule 2) to subsystem A, behaviours x, p, and q; etc.

13.8.2.3.2 Work unit ADV_TDS.2-10

The evaluator ***shall examine*** the TOE design to determine that it is an accurate instantiation of all security functional requirements.

The evaluator ensures that each security requirement listed in the TOE security functional requirements subclause of the ST has a corresponding design description in the TOE design that accurately details how the TSF meets that requirement. This requires that the evaluator identify a collection of subsystems that are responsible for implementing a given functional requirement, and then examine those subsystems to understand how the requirement is implemented. Finally, the evaluator would assess whether the requirement was accurately implemented.

As an example, if the ST requirements specified a role-based access control mechanism, the evaluator would first identify the subsystems that contribute to this mechanism's implementation. This can be done by in-depth knowledge or understanding of the TOE design or by work done in the previous work unit. Note that this trace is only to identify the subsystems, and is not the complete analysis.

The next step would be to understand what mechanism the subsystems implemented. For instance, if the design described an implementation of access control based on UNIX-style protection bits, the design would not be an accurate instantiation of those access control requirements present in the ST example used above. If the evaluator cannot determine that the mechanism was accurately implemented because of a lack of detail, the evaluator would have to assess whether all of the SFR-enforcing subsystems have been identified, or if adequate detail had been provided for those subsystems.

13.8.3 Evaluation of sub-activity (ADV_TDS.3)

13.8.3.1 Objectives

The objective of this sub-activity is to determine whether the TOE design provides a description of the TOE in terms of subsystems sufficient to determine the TSF boundary, and provides a description of the TSF internals in terms of modules (and optionally higher-level abstractions). It provides a detailed description of the SFR-enforcing modules and enough information about the SFR-supporting and SFR-non-interfering modules for the evaluator to determine that the SFRs are completely and accurately implemented; as such, the TOE design provides an explanation of the implementation representation.

13.8.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) security architecture description;

d) the TOE design.

13.8.3.3 Application notes

There are three types of activity that the evaluator must undertake with respect to the TOE design. First, the evaluator determines that the TSF boundary has been adequately described. Second, the evaluator determines that the developer has provided documentation that conforms to the content and presentation requirements for this subsystem, and that is consistent with other documentation provided for the TOE. Finally, the evaluator must analyse the design information provided for the SFR-enforcing modules (at a detailed level) and the SFR-supporting and SFR-non-interfering modules (at a less detailed level) to understand how the system is implemented, and with that knowledge ensure that the TSFI in the functional specification are adequately described, and that the test information adequately tests the TSF (done in the Class ATE: Tests work units).

It is important to note that while the developer is obligated to provide a complete description of the TSF (although SFR-enforcing modules will have more detail than the SFR-supporting or SFR-non-interfering modules), the evaluator is expected to use their judgement in performing their analysis. While the evaluator is expected to look at every module, the detail to which they examine each module may vary. The evaluator analyses each module in order to gain enough understanding to determine the effect of the functionality of the module on the security of the system, and the depth to which they need to analyse the module may vary depending on the module's role in the system. An important aspect of this analysis is that the evaluator should use the other documentation provided (TSS, functional specification, security architecture description, and the TSF internal document) in order to determine that the functionality that is described is correct, and that the implicit designation of SFR-supporting or SFR-non-interfering modules (see below) is supported by their role in the system architecture.

The developer may designate modules as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the modules have been categorised by the developer or not, it is the evaluator's responsibility to determine that the modules have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from

the developer should the developer fail to provide the required information for a particular module.

13.8.3.4 Action ADV_TDS.3.1E

13.8.3.4.1 General

CC Part 3 ADV_TDS.3.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

13.8.3.4.2 Work unit ADV_TDS.3-1

The evaluator ***shall examine*** the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.

The evaluator ensures that all of the subsystems of the TOE are identified. This description of the TOE will be used as input to work unit ADV_TDS.3-2, where the parts of the TOE that make up the TSF are identified. That is, this requirement is on the entire TOE rather than on only the TSF.

The TOE (and TSF) may be described in multiple layers of abstraction (i.e. subsystems and modules). Depending upon the complexity of the TOE, its design may be described in terms of subsystems and modules, as described in CC Part 3, Annex A.4, ADV_TDS: Subsystems and Modules. For a very simple TOE that can be described solely at the "module" level (see ADV_TDS.3-2), this work unit is not applicable and therefore considered to be satisfied.

In performing this activity, the evaluator examines other evidence presented for the TOE (e.g., ST, operator user guidance) to determine that the description of the TOE in such evidence is consistent with the description contained in the TOE design.

CC Part 3 ADV_TDS.3.2C: *The design shall describe the TSF in terms of modules.*

13.8.3.4.3 Work unit ADV_TDS.3-2

The evaluator ***shall examine*** the TOE design to determine that the entire TSF is described in terms of modules.

The evaluator will examine the modules for specific properties in other work units; in this work unit the evaluator determines that the modular description covers the entire TSF, and not just a portion of the TSF. The evaluator uses other evidence provided for the evaluation (e.g., functional specification, security architecture description) in making this determination. For example, if the functional specification contains interfaces to functionality that does not appear to be described in the TOE design description, it may be the case that a portion of the TSF has not been included appropriately. Making this determination will likely be an iterative process, whereas more analysis is done on the other evidence, more confidence can be gained with respect to the completeness of the documentation.

Unlike subsystems, modules describe the implementation in a level of detail that can serve as a guide to reviewing the implementation representation. A description of a module should be such that one can create an implementation of the module from the description, and the resulting implementation would be 1) identical to the actual TSF implementation in terms of the interfaces presented, 2) identical in the use of interfaces that are mentioned in the design, and 3) functionally equivalent to the description of the purpose of the TSF module. For instance, RFC 793 provides a high-level description of the TCP protocol. It is necessarily implementation independent. While it provides a wealth of detail, it is ***not*** a suitable design description because it is not specific to an implementation. An actual implementation can add to the protocol specified in the RFC, and implementation choices (for instance, the use of global data vs. local data in various parts of the implementation) may have an impact on the analysis that is performed. The design description of the TCP module would list the interfaces presented by the implementation

(rather than just those defined in RFC 793), as well as an algorithm description of the processing associated with the modules implementing TCP (assuming it was part of the TSF).

CC Part 3 ADV_TDS.3.3C: *The design shall identify all subsystems of the TSF.*

13.8.3.4.4 Work unit ADV_TDS.3-3

The evaluator ***shall examine*** the TOE design to determine that all subsystems of the TSF are identified.

If the design is presented solely in terms of modules, then subsystems in these requirements are equivalent to modules and the activity should be performed at the module level.

In work unit ADV_TDS.3-1 all of the subsystems of the TOE were identified, and a determination made that the non-TSF subsystems were correctly characterised. Building on that work, the subsystems that were not characterised as non-TSF subsystems should be precisely identified. The evaluator determines that, of the hardware and software installed and configured according to the Preparative procedures (AGD_PRE) guidance, each subsystem has been accounted for as either one that is part of the TSF, or one that is not.

CC Part 3 ADV_TDS.3.4C: *The design shall provide a description of each subsystem of the TSF.*

13.8.3.4.5 Work unit ADV_TDS.3-4

The evaluator ***shall examine*** the TOE design to determine that each subsystem of the TSF describes its role in the enforcement of SFRs described in the ST.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of the subsystem-level description is to give the evaluator context for the modular description that follows. Therefore, the evaluator ensures that the subsystem-level description contains a description of how the security functional requirements are achieved in the design, but at a level of abstraction above the modular description. This description should discuss the mechanisms used at a level that is aligned with the module description; this will provide the evaluators the road map needed to intelligently assess the information contained in the module description. A well-written set of subsystem descriptions will help guide the evaluator in determining the modules that are most important to examine, thus focusing the evaluation activity on the portions of the TSF that have the most relevance with respect to the enforcement of the SFRs.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem plays in enforcing or supporting the implementation of the SFRs, enough information must be present so that a context for understanding the SFR-related functionality is provided.

13.8.3.4.6 Work unit ADV_TDS.3-5

The evaluator ***shall examine*** the TOE design to determine that each SFR-non-interfering subsystem of the TSF is described such that the evaluator can determine that the subsystem is SFR-non-interfering.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

An SFR-non-interfering subsystem is one on which the SFR-enforcing and SFR-supporting subsystems have no dependence; that is, they play no role in implementing SFR functionality.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem does not play in enforcing or supporting the implementation of the SFRs, enough information must be present so that a context for understanding the SFR-non-interfering functionality is provided.

CC Part 3 ADV_TDS.3.5C: *The design shall provide a description of the interactions among all subsystems of the TSF.*

13.8.3.4.7 Work unit ADV_TDS.3-6

The evaluator ***shall examine*** the TOE design to determine that interactions between the subsystems of the TSF are described.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of describing the interactions between the subsystems is to help provide the reader a better understanding of how the TSF performs its functions. These interactions do not need to be characterised at the implementation level (e.g., parameters passed from one routine in a subsystem to a routine in a different subsystem; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular subsystem that are going to be used by another subsystem should be covered in this discussion. Any control relationships between subsystems (e.g., a subsystem responsible for configuring a rule base for a firewall system and the subsystem that actually implements these rules) should also be described.

It should be noted while the developer should characterise all interactions between subsystems, the evaluators need to use their own judgement in assessing the completeness of the description. If the reason for an interaction is unclear, or if there are SFR-related interactions (discovered, for instance, in examining the module-level documentation) that do not appear to be described, the evaluator ensures that this information is provided by the developer. However, if the evaluator can determine that interactions among a particular set of subsystems, while incompletely described by the developer, and a complete description will not aid in understanding the overall functionality nor security functionality provided by the TSF, then the evaluator may choose to consider the description sufficient, and not pursue completeness for its own sake.

CC Part 3 ADV_TDS.3.6C: *The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.*

13.8.3.4.8 Work unit ADV_TDS.3-7

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF and the modules of the TSF is complete.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. To determine completeness, the evaluator examines each mapping and determines that all subsystems map to at least one module, and that all modules map to exactly one subsystem.

13.8.3.4.9 Work unit ADV_TDS.3-8

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF and the modules of the TSF is accurate.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. The evaluator may choose to check the accuracy of the mapping in conjunction with performing other work units. An "inaccurate" mapping is one where the module is mistakenly associated with a subsystem where its functions are not used within the subsystem. Because the mapping is intended to be a guide supporting more detailed analysis, the evaluator is cautioned to apply appropriate effort to this work unit. Expending extensive evaluator resources verifying the accuracy of the mapping is not necessary. Inaccuracies that lead to misunderstandings related to the design that are uncovered as part of this or other work units are the ones that should be associated with this work unit and corrected.

CC Part 3 ADV_TDS.3.7C: *The design shall describe each SFR-enforcing module in terms of its purpose and relationship with other modules.*

13.8.3.4.10 Work unit ADV_TDS.3-9

The evaluator ***shall examine*** the TOE design to determine that the description of the purpose of each SFR-enforcing module and relationship with other modules is complete and accurate.

The developer may designate modules as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the modules have been categorised by the developer or not, it is the evaluator's responsibility to determine that the modules have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular module.

The purpose of a module provides a description indicating what function the module is fulfilling. A word of caution to evaluator is in order. The focus of this work unit should be to provide the evaluator an understanding of how the module works so that determinations can be made about the soundness of the implementation of the SFRs, as well as to support architectural analysis performed for ADV_ARC component. As long as the evaluator has a sound understanding of the module's operation, and its relationship to other modules and the TOE as a whole, the evaluator should consider the objective of the work achieved and not engage in a documentation exercise for the developer (by requiring, for example, a complete algorithmic description for a self-evident implementation representation).

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the TSF internals, or the security architecture description. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the purpose is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.3.10C element, which maps the TSFI in the functional specification to the modules of the TSF.

CC Part 3 ADV_TDS.3.8C: *The design shall describe each SFR-enforcing module in terms of its SFR-related interfaces, return values from those interfaces, interaction with other modules and called SFR-related interfaces to other SFR-enforcing modules.*

13.8.3.4.11 Work unit ADV_TDS.3-10

The evaluator ***shall examine*** the TOE design to determine that the description of the interfaces presented by each SFR-enforcing module contain an accurate and complete description of the SFR-related parameters, the calling conventions for each interface, and any values returned directly by the interface.

The SFR-related interfaces of a module are those interfaces used by other modules as a means to invoke the SFR-related operations provided, and to provide inputs to or receive outputs from the module. The purpose in the specification of these interfaces is to permit the exercise of them during testing. Inter-module interfaces that are not SFR-related need not be specified or described, since they are not a factor in testing. Likewise, other internal interfaces that are not a factor in traversing SFR-related paths of execution (such as those internal paths that are fixed) need not be specified or described, since they are not a factor in testing.

SFR-related interfaces are described in terms of how they are invoked, and any values that are returned. This description would include a list of SFR-related parameters, and descriptions of these parameters. Note that global data would also be considered parameters if used by the module (either as inputs or outputs) when invoked. If a parameter were expected to take on a set of values (e.g., a "flag" parameter), the complete set of values the parameter can take on that would have an effect on module processing would be specified. Likewise, parameters representing data structures are described such that each field of the data structure is identified and described. Note that different programming languages may have additional "interfaces" that would be non-obvious; an example would be operator/function overloading in C++. This "implicit interface" in the class description would also be described as part of the low-level TOE design. Note that although a module can present only one interface, it is more common that a module presents a small set of related interfaces.

In terms of the assessment of parameters (inputs and outputs) to a module, any use of global data must also be considered. A module "uses" global data if it either reads or writes the data. In order to assure the description of such parameters (if used) is complete, the evaluator uses other information provided about the module in the TOE design (interfaces, algorithmic description, etc.), as well as the description of the particular set of global data assessed in work unit ADV_TDS.3-10. For instance, the evaluator can first determine the processing the module performs by examining its function and interfaces presented (particularly the parameters of the interfaces). They can then check to see if the processing appears to "touch" any of the global data areas identified in the TOE design. The evaluator then determines that, for each global data area that appears to be "touched", that global data area is listed as a means of input or output by the module the evaluator is examining.

Invocation conventions are a programming-reference-type description that one can use to correctly invoke a module's interface if one were writing a program to make use of the module's functionality through that interface. This includes necessary inputs and outputs, including any set-up that may need to be performed with respect to global variables.

Values returned through the interface refer to values that are either passed through parameters or messages; values that the function call itself returns in the style of a "C" program function call; or values passed through global means (such as certain error routines in *ix-style operating systems).

In order to assure the description is complete, the evaluator uses other information provided about the module in the TOE design (e.g., algorithmic description, global data used) to ensure that

it appears all data necessary for performing the functions of the module is presented to the module, and that any values that other modules expect the module under examination to provide are identified as being returned by the module. The evaluator determines accuracy by ensuring that the description of the processing matches the information listed as being passed to or from an interface.

CC Part 3 ADV_TDS.3.9C: *The design shall describe each SFR-supporting and SFR-non-interfering module in terms of its purpose and interaction with other modules.*

13.8.3.4.12 Work unit ADV_TDS.3-11

The evaluator ***shall examine*** the TOE design to determine that SFR-supporting and SFR-non-interfering modules are correctly categorised.

In the cases where the developer has provided different amounts of information for different modules, an implicit categorisation has been done. That is, modules (for instance) with detail presented on their SFR-related interfaces (see ADV_TDS.3.10C) are candidate SFR-enforcing modules, although examination by the evaluator may lead to a determination that some set of them are SFR-supporting or SFR-non-interfering. Those with only a description of their purpose and interaction with other modules (for instance) are "implicitly categorised" as SFR-supporting or SFR-non-interfering.

In these cases, a key focus of the evaluator for this work unit is attempting to determine from the evidence provided for each module implicitly categorised as SFR-supporting or SFR-non-interfering and the evaluation information about other modules (in the TOE design, the functional specification, the security architecture description, and the operational user guidance), whether the module is indeed SFR-supporting or SFR-non-interfering. At this level of assurance some error should be tolerated; the evaluator does not have to be absolutely sure that a given module is SFR-supporting or SFR-non-interfering, even though it is labelled as such. However, if the evidence provided indicates that a SFR-supporting or SFR-non-interfering module is SFR-enforcing, the evaluator requests additional information from the developer in order to resolve the apparent inconsistency. For instance, suppose the documentation for Module A (an SFR-enforcing module) indicates that it calls Module B to perform an access check on a certain type of construct. When the evaluator examines the information associated with Module B, they find that all the developer has provided is a purpose and a set of interactions (thus implicitly categorising Module B as SFR-supporting or SFR-non-interfering). On examining the purpose and interactions from Module A, the evaluator finds no mention of Module B performing any access checks, and Module A is not listed as a module with which Module B interacts. At this point the evaluator should approach the developer to resolve the discrepancies between the information provided in Module A and that in Module B.

Another example would be where the evaluator examines the mapping of the TSFI to the modules as provided byADV_TDS.3.2D. This examination shows that Module C is associated with an SFR requiring identification of the user. Again, when the evaluator examines the information associated with Module C, they find that all the developer has provided is a purpose and a set of interactions (thus implicitly categorising Module C as SFR-supporting or SFR-non-interfering). Examining the purpose and interactions presented for Module C, the evaluator is unable to determine why Module C, listed as mapping to a TSFI concerned with user identification, would not be classified as SFR-enforcing. Again, the evaluator should approach the developer to resolve this discrepancy.

A final example is from the opposite point of view. As before, the developer has provided information associated with Module D consisting of a purpose and a set of interactions (thus implicitly categorising Module D as SFR-supporting or SFR-non-interfering). The evaluator examines all of the evidence provided, including the purpose and interactions for Module D. The purpose appears to give a meaningful description of Module D's function in the TOE, the

interactions are consistent with that description, and there is nothing to indicate that Module D is SFR-enforcing. In this case, the evaluator should not demand more information about Module D "just to be sure" it is correctly categorised. The developer has met their obligations and the resulting assurance the evaluator has in the implicit categorisation of Module D is (by definition) appropriate for this assurance level.

13.8.3.4.13 Work unit ADV_TDS.3-12

The evaluator ***shall examine*** the TOE design to determine that the description of the purpose of each SFR-supporting or SFR-non-interfering module is complete and accurate.

The description of the purpose of a module indicates what function the module is fulfilling. From the description, the evaluator should be able to obtain a general idea of the module's role. In order to assure the description is complete, the evaluator uses the information provided about the module's interactions with other modules to assess whether the reasons for the module being called are consistent with the module's purpose. If the interaction description contains functionality that is not apparent from, or in conflict with, the module's purpose, the evaluator needs to determine whether the problem is one of accuracy or of completeness. The evaluator should be wary of purposes that are too short, since meaningful analysis based on a one-sentence purpose is likely to be impossible.

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as administrative guidance, the functional specification, the security architecture description, or the TSF internals document. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the function is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.3.10C element, which maps the TSFI in the functional specification to the modules of the TSF.

13.8.3.4.14 Work unit ADV_TDS.3-13

The evaluator ***shall examine*** the TOE design to determine that the description of an SFR-supporting or SFR-non-interfering module's interaction with other modules is complete and accurate.

It is important to note that, in terms of the Part 3 requirement and this work unit, the term *interaction* is intended to convey less rigour than *interface*. An interaction does not need to be characterised at the implementation level (e.g., parameters passed from one routine in a module to a routine in a different module; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular module that are going to be used by another module should be covered in this discussion. Any control relationships between modules (e.g., a module responsible for configuring a rule base for a firewall system and the module that actually implements these rules) should also be described.

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the security architecture description, or the TSF internals document. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the function is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.3.10C element, which maps the TSFI in the functional specification to the modules of the TSF.

A module's interaction with other modules goes beyond just a call-tree-type document. The interaction is described from a functional perspective of why a module interacts with other modules. The module's purpose describes what functions the module provides to other modules;

the interactions should describe what the module depends on from other modules in order to accomplish this function.

CC Part 3 ADV_TDS.3.10C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

13.8.3.4.15 Work unit ADV_TDS.3-14

The evaluator ***shall examine*** the TOE design to determine that it contains a complete and accurate mapping from the TSFI described in the functional specification to the modules of the TSF described in the TOE design.

The modules described in the TOE design provide a description of the implementation of the TSF. The TSFI provide a description of how the implementation is exercised. The evidence from the developer identifies the module that is initially invoked when an operation is requested at the TSFI, and identifies the chain of modules invoked up to the module that is primarily responsible for implementing the functionality. However, a complete call tree for each TSFI is not required for this work unit. The cases in which more than one module would have to be identified are where there are "entry point" modules or wrapper modules that have no functionality other than conditioning inputs or de-multiplexing an input. Mapping to one of these modules would not provide any useful information to the evaluator.

The evaluator assesses the completeness of the mapping by ensuring that all of the TSFI map to at least one module. The verification of accuracy is more complex.

The first aspect of accuracy is that each TSFI is mapped to a module at the TSF boundary. This determination can be made by reviewing the module description and its interfaces/interactions. The next aspect of accuracy is that each TSFI identifies a chain of modules between the initial module identified and a module that is primarily responsible for implementing the function presented at the TSF. Note that this may be the initial module, or there may be several modules, depending on how much pre-conditioning of the inputs is done. It should be noted that one indicator of a pre-conditioning module is that it is invoked for a large number of the TSFI, where the TSFI are all of similar type (e.g., system call). The final aspect of accuracy is that the mapping makes sense. For instance, mapping a TSFI dealing with access control to a module that checks passwords is not accurate. The evaluator should again use judgement in making this determination. The goal is that this information aids the evaluator in understanding the system and implementation of the SFRs, and ways in which entities at the TSF boundary can interact with the TSF. The bulk of the assessment of whether the SFRs are described accurately by the modules is performed in other work units.

13.8.3.5 Action ADV_TDS.3.2E

13.8.3.5.1 Work unit ADV_TDS.3-15

The evaluator ***shall examine*** the TOE security functional requirements and the TOE design, to determine that all ST security functional requirements are covered by the TOE design.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems, and later to modules. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 Subset access control FDP_ACC.1 Subset access control component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 Subset access control assignment, and these ten rules were implemented in specific places within fifteen modules, it would be inadequate for the evaluator to map FDP_ACC.1 Subset

access control to one subsystem and claim the work unit had been completed. Instead, the evaluator would map FDP_ACC.1 Subset access control (rule 1) to modules x, y, and z of subsystem A; FDP_ACC.1 Subset access control (rule 2) to modules x, p, and q of subsystem A; etc.

13.8.3.5.2 Work unit ADV_TDS.3-16

The evaluator ***shall examine*** the TOE design to determine that it is an accurate instantiation of all security functional requirements.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

As an example, if the ST requirements specified a role-based access control mechanism, the evaluator would first identify the subsystems, and modules that contribute to this mechanism's implementation. This can be done by in-depth knowledge or understanding of the TOE design or by work done in the previous work unit. Note that this trace is only to identify the subsystems, and modules, and is not the complete analysis.

The next step would be to understand what mechanism the subsystems, and modules implemented. For instance, if the design described an implementation of access control based on UNIX-style protection bits, the design would not be an accurate instantiation of those access control requirements present in the ST example used above. If the evaluator cannot determine that the mechanism was accurately implemented because of a lack of detail, the evaluator would have to assess whether all of the SFR-enforcing subsystems and modules have been identified, or if adequate detail had been provided for those subsystems and modules.

13.8.4 Evaluation of sub-activity (ADV_TDS.4)

13.8.4.1 Objectives

The objective of this sub-activity is to determine whether the TOE design provides a description of the TOE in terms of subsystems sufficient to determine the TSF boundary, and provides a description of the TSF internals in terms of modules (and optionally higher-level abstractions). It provides a detailed description of the SFR-enforcing and SFR-supporting modules and enough information about the SFR-non-interfering modules for the evaluator to determine that the SFRs are completely and accurately implemented; as such, the TOE design provides an explanation of the implementation representation.

13.8.4.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) security architecture description;

d) the TOE design.

13.8.4.3 Application notes

There are three types of activity that the evaluator must undertake with respect to the TOE design. First, the evaluator determines that the TSF boundary has been adequately described. Second, the evaluator determines that the developer has provided documentation that conforms to the

content and presentation requirements this subsystem, and that is consistent with other documentation provided for the TOE. Finally, the evaluator must analyse the design information provided for the SFR-enforcing modules (at a detailed level) and the SFR-supporting and SFR-non-interfering modules (at a less detailed level) to understand how the system is implemented, and with that knowledge ensure that the TSFI in the functional specification are adequately described, and that the test information adequately tests the TSF (done in the Class ATE: Tests work units).

13.8.4.4 Action ADV_TDS.4.1E

13.8.4.4.1 General

CC Part 3 ADV_TDS.4.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

13.8.4.4.2 Work unit ADV_TDS.4-1

The evaluator ***shall examine*** the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.

The evaluator ensures that all of the subsystems of the TOE are identified. This description of the TOE will be used as input to work unit ADV_TDS.4-4, where the parts of the TOE that make up the TSF are identified. That is, this requirement is on the entire TOE rather than on only the TSF.

The TOE (and TSF) may be described in multiple layers of abstraction (i.e. subsystems and modules). Depending upon the complexity of the TOE, its design may be described in terms of subsystems and modules, as described in CC Part 3, Annex A.4, ADV_TDS: Subsystems and Modules. For a very simple TOE that can be described solely at the "module" level (see ADV_TDS.4-2), this work unit is not applicable and therefore considered to be satisfied.

In performing this activity, the evaluator examines other evidence presented for the TOE (e.g., ST, operator user guidance) to determine that the description of the TOE in such evidence is consistent with the description contained in the TOE design.

CC Part 3 ADV_TDS.4.2C: *The design shall describe the TSF in terms of modules, designating each module as SFR-enforcing, SFR-supporting, or SFR-non-interfering.*

13.8.4.4.3 Work unit ADV_TDS.4-2

The evaluator ***shall examine*** the TOE design to determine that the entire TSF is described in terms of modules.

The evaluator will examine the modules for specific properties in other work units; in this work unit the evaluator determines that the modular description covers the entire TSF, and not just a portion of the TSF. The evaluator uses other evidence provided for the evaluation (e.g., functional specification, architectural description) in making this determination. For example, if the functional specification contains interfaces to functionality that does not appear to be described in the TOE design description, it may be the case that a portion of the TSF has not been included appropriately. Making this determination will likely be an iterative process, whereas more analysis is done on the other evidence, more confidence can be gained with respect to the completeness of the documentation.

Unlike subsystems, modules describe the implementation in a level of detail that can serve as a guide to reviewing the implementation representation. A description of a module should be such that one can create an implementation of the module from the description, and the resulting implementation would be 1) identical to the actual TSF implementation in terms of the interfaces presented, 2) identical in the use of interfaces that are mentioned in the design, and 3) functionally equivalent to the description of the purpose of the TSF module. For instance, RFC 793

provides a high-level description of the TCP protocol. It is necessarily implementation independent. While it provides a wealth of detail, it is ***not*** a suitable design description because it is not specific to an implementation. An actual implementation can add to the protocol specified in the RFC, and implementation choices (for instance, the use of global data vs. local data in various parts of the implementation) may have an impact on the analysis that is performed. The design description of the TCP module would list the interfaces presented by the implementation (rather than just those defined in RFC 793), as well as an algorithm description of the processing associated with the modules implementing TCP (assuming it was part of the TSF).

13.8.4.4.4 Work unit ADV_TDS.4-3

The evaluator ***shall check*** the TOE design to determine that the TSF modules are identified as either SFR-enforcing, SFR-supporting, or SFR-non-interfering.

The purpose of designating each module (according to the role a particular module plays in the enforcement of the SFRs) is to allow developers to provide less information about the parts of the TSF that have little role in security. It is always permissible for the developer to provide more information or detail than the requirements demand, as can occur when the information has been gathered outside the evaluation context. In such cases the developer must still designate the modules as either SFR-enforcing, SFR-supporting, or SFR-non-interfering.

The accuracy of these designations is continuously reviewed as the evaluation progresses. The concern is the mis-designation of modules as being less important (and hence, having less information) than is really the case. While blatant mis-designations may be immediately apparent (e.g., designating an authentication module as anything but SFR-enforcing when User identification (FIA_UID) is one of the SFRs being claimed), other mis-designations might not be discovered until the TSF is better understood. The evaluator must therefore keep in mind that these designations are the developer's initial best effort, but are subject to change. Further guidance is provided under work unit ADV_TDS.4-17, which examines the accuracy of these designations.

CC Part 3 ADV_TDS.4.3C: *The design shall identify all subsystems of the TSF.*

13.8.4.4.5 Work unit ADV_TDS.4-4

The evaluator ***shall examine*** the TOE design to determine that all subsystems of the TSF are identified.

If the design is presented solely in terms of modules, then subsystems in these requirements are equivalent to modules and the activity should be performed at the module level.

In work unit ADV_TDS.4-1 all of the subsystems of the TOE were identified, and a determination made that the non-TSF subsystems were correctly characterised. Building on that work, the subsystems that were not characterised as non-TSF subsystems should be precisely identified. The evaluator determines that, of the hardware and software installed and configured according to the Preparative procedures (AGD_PRE) guidance, each subsystem has been accounted for as either one that is part of the TSF, or one that is not.

CC Part 3 ADV_TDS.4.4C: *The design shall provide a semiformal description of each subsystem of the TSF, supported by informal, explanatory text where appropriate.*

13.8.4.4.6 Work unit ADV_TDS.4-5

The evaluator ***shall examine*** the TDS documentation to determine that the semiformal notation used for describing the subsystems, modules and their interfaces is defined or referenced.

A semiformal notation can be either defined by the sponsor or a corresponding standard be referenced. The evaluator should provide a mapping of security functions and their interfaces outlining in what part of the documentation a function or interface is semiformal described and what notation is used. The evaluator examines all semiformal notations used to make sure that they are of a semiformal style and to justify the appropriateness of the manner how the semiformal notations are used for the TOE.

The evaluator is reminded that a semi-formal presentation is characterised by a standardised format with a well-defined syntax that reduces ambiguity that may occur in informal presentations. The syntax of all semiformal notations used in the functional specification shall be defined or a corresponding standard be referenced. The evaluator verifies that the semiformal notations used for expressing the functional specification are capable of expressing features relevant to security. In order to determine this, the evaluator can refer to the SFR and compare the TSF security features stated in the ST and those described in the FSP using the semiformal notations.

13.8.4.4.7 Work unit ADV_TDS.4-6

The evaluator ***shall examine*** the TOE design to determine that each subsystem of the TSF describes its role in the enforcement of SFRs described in the ST.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of the subsystem-level description is to give the evaluator context for the modular description that follows. Therefore, the evaluator ensures that the subsystem-level description contains a description of how the security functional requirements are achieved in the design, but at a level of abstraction above the modular description. This description should discuss the mechanisms used at a level that is aligned with the module description; this will provide the evaluators the road map needed to intelligently assess the information contained in the module description. A well-written set of subsystem descriptions will help guide the evaluator in determining the modules that are most important to examine, thus focusing the evaluation activity on the portions of the TSF that have the most relevance with respect to the enforcement of the SFRs.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem plays in enforcing or supporting the implementation of the SFRs, enough information must be present so that a context for understanding the SFR-related functionality is provided.

13.8.4.4.8 Work unit ADV_TDS.4-7

The evaluator ***shall examine*** the TOE design to determine that each SFR-non-interfering subsystem of the TSF is described such that the evaluator can determine that the subsystem is SFR-non-interfering.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

An SFR-non-interfering subsystem is one on which the SFR-enforcing and SFR-supporting subsystems have no dependence; that is, they play no role in implementing SFR functionality.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem does not play in enforcing or supporting the

implementation of the SFRs, enough information must be present so that a context for understanding the SFR-non-interfering functionality is provided.

CC Part 3 ADV_TDS.4.5C: *The design shall provide a description of the interactions among all subsystems of the TSF.*

13.8.4.4.9 Work unit ADV_TDS.4-8

The evaluator ***shall examine*** the TOE design to determine that interactions between the subsystems of the TSF are described.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of describing the interactions between the subsystems is to help provide the reader a better understanding of how the TSF performs it functions. These interactions do not need to be characterised at the implementation level (e.g., parameters passed from one routine in a subsystem to a routine in a different subsystem; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular subsystem that are going to be used by another subsystem need to be covered in this discussion. Any control relationships between subsystems (e.g., a subsystem responsible for configuring a rule base for a firewall system and the subsystem that actually implements these rules) should also be described.

It should be noted while the developer should characterise all interactions between subsystems, the evaluators need to use their own judgement in assessing the completeness of the description. If the reason for an interaction is unclear, or if there are SFR-related interactions (discovered, for instance, in examining the module-level documentation) that do not appear to be described, the evaluator ensures that this information is provided by the developer. However, if the evaluator can determine that interactions among a particular set of subsystems, while incompletely described by the developer, and a complete description will not aid in understanding the overall functionality nor security functionality provided by the TSF, then the evaluator may choose to consider the description sufficient, and not pursue completeness for its own sake.

CC Part 3 ADV_TDS.4.6C: *The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.*

13.8.4.4.10 Work unit ADV_TDS.4-9

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF and the modules of the TSF is complete.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. To determine completeness, the evaluator examines each mapping and determines that all subsystems map to at least one module, and that all modules map to exactly one subsystem.

13.8.4.4.11 Work unit ADV_TDS.4-10

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF to the modules of the TSF is accurate.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. The evaluator may choose to check the accuracy of the mapping in conjunction with performing other work units. An "inaccurate" mapping is one where the module is mistakenly associated with a subsystem where its functions are not used within the subsystem. Because the mapping is intended to be a guide supporting more detailed analysis, the evaluator is cautioned to apply appropriate effort to this work unit. Expending extensive evaluator resources verifying the accuracy of the mapping is not necessary. Inaccuracies that lead to misunderstandings related to the design that are uncovered as part of this or other work units are the ones that should be associated with this work unit and corrected.

CC Part 3 ADV_TDS.4.7C: *The design shall describe each SFR-enforcing and SFR-supporting module in terms of its purpose and relationship with other modules.*

13.8.4.4.12 Work unit ADV_TDS.4-11

The evaluator ***shall examine*** the TOE design to determine that the description of the purpose of each SFR-enforcing and SFR-supporting module, and relationship with other modules is complete and accurate.

The developer may designate modules as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the modules have been categorised by the developer or not, it is the evaluator's responsibility to determine that the modules have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular module.

The purpose of a module provides a description indicating what function the module is fulfilling. A word of caution to evaluator is in order. The focus of this work unit should be to provide the evaluator an understanding of how the module works so that determinations can be made about the soundness of the implementation of the SFRs, as well as to support architectural analysis performed for ADV_ARC subsystems. As long as the evaluator has a sound understanding of the module's operation, and its relationship to other modules and the TOE as a whole, the evaluator should consider the objective of the work achieved and not engage in a documentation exercise for the developer (by requiring, for example, a complete algorithmic description for a self-evident implementation representation).

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the TSF internals, or the security architecture description. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the purpose is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.4.10C element, which maps the TSFI in the functional specification to the modules of the TSF.

CC Part 3 ADV_TDS.4.8C: *The design shall describe each SFR-enforcing and SFR-supporting module in terms of its SFR-related interfaces, return values from those interfaces, interaction with other modules and called SFR-related interfaces to other SFR-enforcing or SFR-supporting modules.*

13.8.4.4.13 Work unit ADV_TDS.4-12

The evaluator ***shall examine*** the TOE design to determine that the description of the interfaces presented by each SFR-enforcing and SFR-supporting module contain an accurate and complete description of the SFR-related parameters, the invocation conventions for each interface, and any values returned directly by the interface.

The SFR-related interfaces of a module are those interfaces used by other modules as a means to invoke the SFR-related operations provided, and to provide inputs to or receive outputs from the module. The purpose in the specification of these interfaces is to permit the exercise of them during testing. Inter-module interfaces that are not SFR-related need not be specified or described, since they are not a factor in testing. Likewise, other internal interfaces that are not a factor in traversing SFR-related paths of execution (such as those internal paths that are fixed).

SFR-related interfaces of SFR-supporting modules are all interfaces of SFR-supporting modules that are called directly or indirectly from SFR-enforcing modules. Those interfaces need to be described with all the parameter used in such a call. This allows the evaluator to understand the purpose of the call to the SFR-supporting module in the context of operation of the SFR-enforcing modules.

SFR-related interfaces are described in terms of how they are invoked, and any values that are returned. This description would include a list of parameters, and descriptions of these parameters. Note that global data would also be considered parameters if used by the module (either as inputs or outputs) when invoked. If a parameter were expected to take on a set of values (e.g., a "flag" parameter), the complete set of values the parameter can take on that would have an effect on module processing would be specified. Likewise, parameters representing data structures are described such that each field of the data structure is identified and described. Note that different programming languages may have additional "interfaces" that would be non-obvious; an example would be operator/function overloading in C++. This "implicit interface" in the class description would also be described as part of the low-level TOE design. Note that although a module can present only one interface, it is more common that a module presents a small set of related interfaces.

In terms of the assessment of parameters (inputs and outputs) to a module, any use of global data must also be considered. A module "uses" global data if it either reads or writes the data. In order to assure the description of such parameters (if used) is complete, the evaluator uses other information provided about the module in the TOE design (interfaces, algorithmic description, etc.), as well as the description of the particular set of global data assessed in work unit ADV_TDS.4-12. For instance, the evaluator can first determine the processing the module performs by examining its function and interfaces presented (particularly the parameters of the interfaces). They can then check to see if the processing appears to "touch" any of the global data areas identified in the TDS design. The evaluator then determines that, for each global data area that appears to be "touched", that global data area is listed as a means of input or output by the module the evaluator is examining.

Invocation conventions are a programming-reference-type description that one can use to correctly invoke a module's interface if one were writing a program to make use of the module's functionality through that interface. This includes necessary inputs and outputs, including any set-up that may need to be performed with respect to global variables.

Values returned through the interface refer to values that are either passed through parameters or messages; values that the function call itself returns in the style of a "C" program function call; or values passed through global means (such as certain error routines in *ix-style operating systems).

In order to assure the description is complete, the evaluator uses other information provided about the module in the TOE design (e.g., algorithmic description, global data used) to ensure that it appears all data necessary for performing the functions of the module is presented to the module, and that any values that other modules expect the module under examination to provide are identified as being returned by the module. The evaluator determines accuracy by ensuring that the description of the processing matches the information listed as being passed to or from an interface.

CC Part 3 ADV_TDS.4.9C: *The design shall describe each SFR-non-interfering module in terms of its purpose and interaction with other modules.*

13.8.4.4.14 Work unit ADV_TDS.4-13

The evaluator ***shall examine*** the TOE design to determine that SFR-non-interfering modules are correctly categorised.

As mentioned in work unit ADV_TDS.4-2, less information is required about modules that are SFR-non-interfering. A key focus of the evaluator for this work unit is attempting to determine from the evidence provided for each module implicitly categorised as SFR-non-interfering and the evaluation (information about other modules in the TOE design, the functional specification, the security architecture description, the operational user guidance, the TSF internals document, and perhaps even the implementation representation) whether the module is indeed SFR-non-interfering. At this level of assurance some error should be tolerated; the evaluator does not have to be absolutely sure that a given module is SFR-non-interfering, even though it is labelled as such. However, if the evidence provided indicates that a SFR-non-interfering module is SFR-enforcing or SFR-supporting, the evaluator requests additional information from the developer in order to resolve the apparent inconsistency. For example, suppose the documentation for Module A (an SFR-enforcing module) indicates that it calls Module B to perform an access check on a certain type of construct. When the evaluator examines the information associated with Module B, it is discovered that the only information the developer has provided is a purpose and a set of interactions (thus implicitly categorising Module B as SFR-supporting or SFR-non-interfering). On examining the purpose and interactions from Module A, the evaluator finds no mention of Module B performing any access checks, and Module A is not listed as a module with which Module B interacts. At this point the evaluator should approach the developer to resolve the discrepancies between the information provided in Module A and that in Module B.

Another example would be where the evaluator examines the mapping of the TSFI to the modules as provided by ADV_TDS.4.2D. This examination shows that Module C is associated with an SFR requiring identification of the user. Again, when the evaluator examines the information associated with Module C, they find that all the developer has provided is a purpose and a set of interactions (thus implicitly categorising Module C as SFR-non-interfering). Examining the purpose and interactions presented for Module C, the evaluator is unable to determine why Module C, listed as mapping to a TSFI concerned with user identification, would not be classified as SFR-enforcing or SFR-supporting. Again, the evaluator should approach the developer to resolve this discrepancy.

A final example illustrates the opposite situation. As before, the developer has provided information associated with Module D consisting of a purpose and a set of interactions (thus implicitly categorising Module D as SFR-non-interfering). The evaluator examines all of the evidence provided, including the purpose and interactions for Module D. The purpose appears to give a meaningful description of Module D's function in the TOE, the interactions are consistent with that description, and there is nothing to indicate that Module D is SFR-enforcing or SFR-supporting. In this case, the evaluator should not demand more information about Module D "just to be sure" it is correctly categorised. The developer has met the obligations and the resulting

assurance the evaluator has in the implicit categorisation of Module D is (by definition) appropriate for this assurance level.

13.8.4.4.15 Work unit ADV_TDS.4-14

The evaluator ***shall examine*** the TOE design to determine that the description of the purpose of each SFR-non-interfering module is complete and accurate.

The description of the purpose of a module indicates what function the module is fulfilling. From the description, the evaluator should be able to obtain a general idea of the module's role. In order to assure the description is complete, the evaluator uses the information provided about the module's interactions with other modules to assess whether the reasons for the module being called are consistent with the module's purpose. If the interaction description contains functionality that is not apparent from, or in conflict with, the module's purpose, the evaluator needs to determine whether the problem is one of accuracy or of completeness. The evaluator should be wary of purposes that are too short, since meaningful analysis based on a one-sentence purpose is likely to be impossible.

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the security architecture description, or the TSF internals document. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the function is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.4.10C element, which maps the TSFI in the functional specification to the modules of the TSF.

13.8.4.4.16 Work unit ADV_TDS.4-15

The evaluator ***shall examine*** the TOE design to determine that the description of an SFR-non-interfering module's interaction with other modules is complete and accurate.

It is important to note that, in terms of the Part 3 requirement and this work unit, the term *interaction* is intended to convey less rigour than *interface*. An interaction does not need to be characterised at the implementation level (e.g., parameters passed from one routine in a module to a routine in a different module; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular module that are going to be used by another module should be covered in this discussion. Any control relationships between modules (e.g., a module responsible for configuring a rule base for a firewall system and the module that actually implements these rules) should also be described.

A module's interaction with other modules can be captured in many ways. The intent for the TOE design is to allow the evaluator to understand (in part through analysis of module interactions) the role of the SFR-supporting and SFR-non-interfering modules in the overall TOE design. Understanding of this role will aid the evaluator in performing work unit ADV_TDS.4-8.

A module's interaction with other modules goes beyond just a call-tree-type document. The interaction is described from a functional perspective of why a module interacts with other modules. The module's purpose describes what functions the module provides to other modules; the interactions should describe what the module depends on from other modules in order to accomplish this function.

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the security architecture description, or the TSF internals document. However, the

evaluator uses the information present in those documents to the extent possible to help ensure that the interactions are accurately and completely described.

CC Part 3 ADV_TDS.4.10C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

13.8.4.4.17 Work unit ADV_TDS.4-16

The evaluator ***shall examine*** the TOE design to determine that it contains a complete and accurate mapping from the TSFI described in the functional specification to the modules of the TSF described in the TOE design.

The modules described in the TOE design provide a description of the implementation of the TSF. The TSFI provide a description of how the implementation is exercised. The evidence from the developer identifies the module that is initially invoked when an operation is requested at the TSFI, and identify the chain of modules invoked up to the module that is primarily responsible for implementing the functionality. However, a complete call tree for each TSFI is not required for this work unit. The cases in which more than one module would have to be identified are where there are "entry point" modules or wrapper modules that have no functionality other than conditioning inputs or de-multiplexing an input. Mapping to one of these modules would not provide any useful information to the evaluator.

The evaluator assesses the completeness of the mapping by ensuring that all of the TSFI map to at least one module. The verification of accuracy is more complex.

The first aspect of accuracy is that each TSFI is mapped to a module at the TSF boundary. This determination can be made by reviewing the module description and its interfaces/interactions. The next aspect of accuracy is that each TSFI identifies a chain of modules between the initial module identified and a module that is primarily responsible for implementing the function presented at the TSF. Note that this may be the initial module, or there may be several modules, depending on how much pre-conditioning of the inputs is done. It should be noted that one indicator of a pre-conditioning module is that it is invoked for a large number of the TSFI, where the TSFI are all of similar type (e.g., system call). The final aspect of accuracy is that the mapping makes sense. For instance, mapping a TSFI dealing with access control to a module that checks passwords is not accurate. The evaluator should again use judgement in making this determination. The goal is that this information aids the evaluator in understanding the system and implementation of the SFRs, and ways in which entities at the TSF boundary can interact with the TSF. The bulk of the assessment of whether the SFRs are described accurately by the modules is performed in other work units.

13.8.4.5 Action ADV_TDS.4.2E

13.8.4.5.1 Work unit ADV_TDS.4-17

The evaluator ***shall examine*** the TOE security functional requirements and the TOE design, to determine that all ST security functional requirements are covered by the TOE design.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems, and later to modules. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 Subset access control component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 Subset access control assignment, and these ten rules were implemented in specific places within fifteen modules, it would be inadequate for the evaluator to map FDP_ACC.1 Subset access control to one subsystem

and claim the work unit had been completed. Instead, the evaluator would map FDP_ACC.1 Subset access control (rule 1) to modules x, y and z of subsystem A; FDP_ACC.1 Subset access control (rule 2) to x, p, and q of subsystem A; etc.

13.8.4.5.2 Work unit ADV_TDS.4-18

The evaluator ***shall examine*** the TOE design to determine that it is an accurate instantiation of all security functional requirements.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

As an example, if the ST requirements specified a role-based access control mechanism, the evaluator would first identify the subsystems, and modules that contribute to this mechanism's implementation. This can be done by in-depth knowledge or understanding of the TOE design or by work done in the previous work unit. Note that this trace is only to identify the subsystems, and modules, and is not the complete analysis.

The next step would be to understand what mechanism the subsystems, and modules implemented. For instance, if the design described an implementation of access control based on UNIX-style protection bits, the design would not be an accurate instantiation of those access control requirements present in the ST example used above. If the evaluator cannot determine that the mechanism was accurately implemented because of a lack of detail, the evaluator would have to assess whether all of the SFR-enforcing subsystems and modules have been identified, or if adequate detail had been provided for those subsystems and modules.

13.8.5 Evaluation of sub-activity (ADV_TDS.5)

13.8.5.1 Objectives

The objectives of this sub-activity are to determine whether the TOE design provides a description of the TOE in terms of subsystems sufficient to determine the TSF boundary, and provides a description of the TSF internals in terms of modules (and optionally higher-level abstractions). It provides enough information about the modules for the evaluator to determine that the SFRs are completely and accurately implemented; as such, the TOE design provides an explanation of the implementation representation.

13.8.5.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) security architecture description;

d) the TOE design.

13.8.5.3 Application notes

There are three types of activity that the evaluator must undertake with respect to the TOE design. First, the evaluator determines that the TSF boundary has been adequately described. Second, the evaluator determines that the developer has provided documentation that conforms to the content and presentation requirements this subsystem, and that is consistent with other

documentation provided for the TOE. Finally, the evaluator must analyse the design information provided for the modules (at a detailed level) to understand how the system is implemented, and with that knowledge ensure that the TSFI in the functional specification are adequately described, and that the test information adequately tests the TSF (done in the Class ATE: Tests work units).

13.8.5.4 Action ADV_TDS.5.1E

13.8.5.4.1 General

CC Part 3 ADV_TDS.5.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

13.8.5.4.2 Work unit ADV_TDS.5-1

The evaluator ***shall examine*** the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.

The evaluator ensures that all of the subsystems of the TOE are identified. This description of the TOE will be used as input to work unit ADV_TDS.5-4, where the parts of the TOE that make up the TSF are identified. That is, this requirement is on the entire TOE rather than on only the TSF.

The TOE (and TSF) may be described in multiple layers of abstraction (i.e. subsystems and modules). Depending upon the complexity of the TOE, its design may be described in terms of subsystems and modules, as described in CC Part 3, Annex A.4, ADV_TDS: Subsystems and Modules. For a very simple TOE that can be described solely at the "module" level (see ADV_TDS.5-2), this work unit is not applicable and therefore considered to be satisfied.

In performing this activity, the evaluator examines other evidence presented for the TOE (e.g., ST, operator user guidance) to determine that the description of the TOE in such evidence is consistent with the description contained in the TOE design.

CC Part 3 ADV_TDS.5.2C: *The design shall describe the TSF in terms of modules, designating each module as SFR-enforcing, SFR-supporting, or SFR-non-interfering.*

13.8.5.4.3 Work unit ADV_TDS.5-2

The evaluator ***shall examine*** the TOE design to determine that the entire TSF is described in terms of modules.

The evaluator will examine the modules for specific properties in other work units; in this work unit the evaluator determines that the modular description covers the entire TSF, and not just a portion of the TSF. The evaluator uses other evidence provided for the evaluation (e.g., functional specification, architectural description) in making this determination. For example, if the functional specification contains interfaces to functionality that does not appear to be described in the TOE design description, it may be the case that a portion of the TSF has not been included appropriately. Making this determination will likely be an iterative process, whereas more analysis is done on the other evidence, more confidence can be gained with respect to the completeness of the documentation.

Unlike subsystems, modules describe the implementation in a level of detail that can serve as a guide to reviewing the implementation representation. A description of a module should be such that one can create an implementation of the module from the description, and the resulting implementation would be 1) identical to the actual TSF implementation in terms of the interfaces presented, 2) identical in the use of interfaces that are mentioned in the design, and 3) functionally equivalent to the description of the purpose of the TSF module. For instance, RFC 793 provides a high-level description of the TCP protocol. It is necessarily implementation independent. While it provides a wealth of detail, it is **not** a suitable design description because it is not specific to an implementation. An actual implementation can add to the protocol specified

in the RFC, and implementation choices (for instance, the use of global data vs. local data in various parts of the implementation) may have an impact on the analysis that is performed. The design description of the TCP module would list the interfaces presented by the implementation (rather than just those defined in RFC 793), as well as an algorithm description of the processing associated with the modules implementing TCP (assuming it was part of the TSF).

13.8.5.4.4 Work unit ADV_TDS.5-3

The evaluator ***shall check*** the TOE design to determine that the TSF modules are identified as either SFR-enforcing, SFR-supporting, or SFR-non-interfering.

The purpose of designating each module (according to the role a particular module plays in the enforcement of the SFRs) is to allow developers to provide less information about the parts of the TSF that have little role in security. It is always permissible for the developer to provide more information or detail than the requirements demand, as can occur when the information has been gathered outside the evaluation context. In such cases the developer must still designate the modules as either SFR-enforcing, SFR-supporting, or SFR-non-interfering.

The accuracy of these designations is continuously reviewed as the evaluation progresses. The concern is the mis-designation of modules as being less important (and hence, having less information) than is really the case. While blatant mis-designations may be immediately apparent (e.g., designating an authentication module as anything but SFR-enforcing when User identification (FIA_UID) is one of the SFRs being claimed), other mis-designations might not be discovered until the TSF is better understood. The evaluator must therefore keep in mind that these designations are the developer's initial best effort, but are subject to change. Further guidance is provided under work unit ADV_TDS.5-16, which examines the accuracy of these designations.

CC Part 3 ADV_TDS.5.3C: *The design shall identify all subsystems of the TSF.*

13.8.5.4.5 Work unit ADV_TDS.5-4

The evaluator ***shall examine*** the TOE design to determine that all subsystems of the TSF are identified.

If the design is presented solely in terms of modules, then subsystems in these requirements are equivalent to modules and the activity should be performed at the module level.

In work unit ADV_TDS.5-1 all of the subsystems of the TOE were identified, and a determination made that the non-TSF subsystems were correctly characterised. Building on that work, the subsystems that were not characterised as non-TSF subsystems should be precisely identified. The evaluator determines that, of the hardware and software installed and configured according to the Preparative procedures (AGD_PRE) guidance, each subsystem has been accounted for as either one that is part of the TSF, or one that is not.

CC Part 3 ADV_TDS.5.4C: *The design shall provide a semiformal description of each subsystem of the TSF, supported by informal, explanatory text where appropriate.*

13.8.5.4.6 Work unit ADV_TDS.5-5

The evaluator ***shall examine*** the TDS documentation to determine that the semiformal notation used for describing the subsystems, modules and their interfaces is defined or referenced.

A semiformal notation can be either defined by the sponsor or a corresponding standard be referenced. The evaluator should provide a mapping of security functions and their interfaces outlining in what part of the documentation a function or interface is semiformal described and what notation is used. The evaluator examines all semiformal notations used to make sure that

they are of a semiformal style and to justify the appropriateness of the manner how the semiformal notations are used for the TOE.

The evaluator is reminded that a semi-formal presentation is characterised by a standardised format with a well-defined syntax that reduces ambiguity that may occur in informal presentations. The syntax of all semiformal notations used in the functional specification shall be defined or a corresponding standard be referenced. The evaluator verifies that the semiformal notations used for expressing the functional specification are capable of expressing features relevant to security. In order to determine this, the evaluator can refer to the SFR and compare the TSF security features stated in the ST and those described in the FSP using the semiformal notations.

Note that ADV_TDS.5.7C requires the module description to be semiformal. This work unit therefore applies also to that description.

13.8.5.4.7 Work unit ADV_TDS.5-6

The evaluator ***shall examine*** the TOE design to determine that each subsystem of the TSF describes its role in the enforcement of SFRs described in the ST.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of the subsystem-level description is to give the evaluator context for the modular description that follows. Therefore, the evaluator ensures that the subsystem-level description contains a description of how the security functional requirements are achieved in the design, but at a level of abstraction above the modular description. This description should discuss the mechanisms used at a level that is aligned with the module description; this will provide the evaluators the road map needed to intelligently assess the information contained in the module description. A well-written set of subsystem descriptions will help guide the evaluator in determining the modules that are most important to examine, thus focusing the evaluation activity on the portions of the TSF that have the most relevance with respect to the enforcement of the SFRs.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem plays in enforcing or supporting the implementation of the SFRs, enough information must be present so that a context for understanding the SFR-related functionality is provided.

13.8.5.4.8 Work unit ADV_TDS.5-7

The evaluator ***shall examine*** the TOE design to determine that each SFR-non-interfering subsystem of the TSF is described such that the evaluator can determine that the subsystem is SFR-non-interfering.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

An SFR-non-interfering subsystem is one on which the SFR-enforcing and SFR-supporting subsystems have no dependence; that is, they play no role in implementing SFR functionality.

The evaluator ensures that all subsystems of the TSF have a description. While the description should focus on the role that the subsystem does not play in enforcing or supporting the

implementation of the SFRs, enough information must be present so that a context for understanding the SFR-non-interfering functionality is provided.

CC Part 3 ADV_TDS.5.5C: *The design shall provide a description of the interactions among all subsystems of the TSF.*

13.8.5.4.9 Work unit ADV_TDS.5-8

The evaluator ***shall examine*** the TOE design to determine that interactions between the subsystems of the TSF are described.

If the design is presented solely in terms of modules, then this work unit will be considered satisfied by the assessment done in subsequent work units; no explicit action on the part of the evaluator is necessary in this case.

On systems that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the goal of describing the interactions between the subsystems is to help provide the reader a better understanding of how the TSF performs it functions. These interactions do not need to be characterised at the implementation level (e.g., parameters passed from one routine in a subsystem to a routine in a different subsystem; global variables; hardware signals (e.g., interrupts) from a hardware subsystem to an interrupt-handling subsystem), but the data elements identified for a particular subsystem that are going to be used by another subsystem need to be covered in this discussion. Any control relationships between subsystems (e.g., a subsystem responsible for configuring a rule base for a firewall system and the subsystem that actually implements these rules) should also be described.

It should be noted while the developer should characterise all interactions between subsystems, the evaluators need to use their own judgement in assessing the completeness of the description. If the reason for an interaction is unclear, or if there are SFR-related interactions (discovered, for instance, in examining the module-level documentation) that do not appear to be described, the evaluator ensures that this information is provided by the developer. However, if the evaluator can determine that interactions among a particular set of subsystems, while incompletely described by the developer, and a complete description will not aid in understanding the overall functionality nor security functionality provided by the TSF, then the evaluator may choose to consider the description sufficient, and not pursue completeness for its own sake.

CC Part 3 ADV_TDS.5.6C: *The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.*

13.8.5.4.10 Work unit ADV_TDS.5-9

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF and the modules of the TSF is complete.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. To determine completeness, the evaluator examines each mapping and determines that all subsystems map to at least one module, and that all modules map to exactly one subsystem.

13.8.5.4.11 Work unit ADV_TDS.5-10

The evaluator ***shall examine*** the TOE design to determine that the mapping between the subsystems of the TSF to the modules of the TSF is accurate.

If the design is presented solely in terms of modules, then this work unit is considered satisfied.

For TOEs that are complex enough to warrant a subsystem-level description of the TSF in addition to the modular description, the developer provides a simple mapping showing how the modules of the TSF are allocated to the subsystems. This will provide the evaluator a guide in performing their module-level assessment. The evaluator may choose to check the accuracy of the mapping in conjunction with performing other work units. An "inaccurate" mapping is one where the module is mistakenly associated with a subsystem where its functions are not used within the subsystem. Because the mapping is intended to be a guide supporting more detailed analysis, the evaluator is cautioned to apply appropriate effort to this work unit. Expending extensive evaluator resources verifying the accuracy of the mapping is not necessary. Inaccuracies that lead to misunderstandings related to the design that are uncovered as part of this or other work units are the ones that should be associated with this work unit and corrected.

CC Part 3 ADV_TDS.5.7C: *The design shall provide a semiformal description of each module in terms of its purpose, interaction, interfaces, return values from those interfaces, and called interfaces to other modules, supported by informal, explanatory text where appropriate.*

13.8.5.4.12 Work unit ADV_TDS.5-11

The evaluator ***shall examine*** the TOE design to determine that the semiformal description of the purpose of each module, and its relationship with other modules is complete and accurate.

The developer may designate modules as SFR-enforcing, SFR-supporting, and SFR non-interfering, but these "tags" are used only to describe the amount and type of information the developer must provide, and can be used to limit the amount of information the developer has to develop if their engineering process does not produce the documentation required. Whether the modules have been categorised by the developer or not, it is the evaluator's responsibility to determine that the modules have the appropriate information for their role (SFR-enforcing, SFR-supporting, or SFR non-interfering) in the TOE, and to obtain the appropriate information from the developer should the developer fail to provide the required information for a particular module.

The purpose of a module provides a description indicating what function the module is fulfilling. A word of caution to the evaluator is in order. The focus of this work unit should be to provide the evaluator an understanding of how the module works so that determinations can be made about the soundness of the implementation of the SFRs, as well as to support architectural analysis performed for ADV_ARC subsystems. As long as the evaluator has a sound understanding of the module's operation, and its relationship to other modules and the TOE as a whole, the evaluator should consider the objective of the work achieved and not engage in a documentation exercise for the developer (by requiring, for example, a complete algorithmic description for a self-evident implementation representation).

Because the modules are at such a low level, it may be difficult determine completeness and accuracy impacts from other documentation, such as operational user guidance, the functional specification, the TSF internals, or the security architecture description. However, the evaluator uses the information present in those documents to the extent possible to help ensure that the purpose is accurately and completely described. This analysis can be aided by the analysis performed for the work units for the ADV_TDS.5.8C element, which maps the TSFI in the functional specification to the modules of the TSF.

13.8.5.4.13 Work unit ADV_TDS.5-12

The evaluator ***shall examine*** the TOE design to determine that the semiformal description of the interfaces presented by each module contain an accurate and complete description of the related

parameters, the invocation conventions for each interface, and any values returned directly by the interface.

The interfaces of a module are those interfaces used by other modules as a means to invoke the operations provided, and to provide inputs to or receive outputs from the module. The purpose in the specification of these interfaces is to permit the exercise of them during testing. Inter-module interfaces that are not SFR-related need not be specified or described, since they are not a factor in testing. Likewise, other internal interfaces that are not a factor in traversing SFR-related paths of execution (such as those internal paths that are fixed).

SFR-related interfaces are all interfaces that are called directly or indirectly from SFR-enforcing modules. Those interfaces need to be described with all the parameter used in such a call. This allows the evaluator to understand the purpose of the call in the context of operation of the SFR-enforcing modules.

SFR-related interfaces are described in terms of how they are invoked, and any values that are returned. This description would include a list of parameters, and descriptions of these parameters. Note that global data would also be considered parameters if used by the module (either as inputs or outputs) when invoked. If a parameter were expected to take on a set of values (e.g., a "flag" parameter), the complete set of values the parameter can take on, that would have an effect on module processing, would be specified. Likewise, parameters representing data structures are described such that each field of the data structure is identified and described. Note that different programming languages may have additional "interfaces" that would be non-obvious; an example would be operator/function overloading in C++. This "implicit interface" in the class description would also be described as part of the low-level TOE design. Note that although a module can present only one interface, it is more common that a module presents a small set of related interfaces.

In terms of the assessment of parameters (inputs and outputs) to a module, any use of global data must also be considered. A module "uses" global data if it either reads or writes the data. In order to assure the description of such parameters (if used) is complete, the evaluator uses other information provided about the module in the TOE design (interfaces, algorithmic description, etc.), as well as the description of the particular set of global data assessed in work unit ADV_TDS.5-10. For instance, the evaluator can first determine the processing the module performs by examining its function and interfaces presented (particularly the parameters of the interfaces). They can then check to see if the processing appears to "touch" any of the global data areas identified in the TDS design. The evaluator then determines that, for each global data area that appears to be "touched", that global data area is listed as a means of input or output by the module the evaluator is examining.

Invocation conventions are a programming-reference-type description that one can use to correctly invoke a module's interface if one were writing a program to make use of the module's functionality through that interface. This includes necessary inputs and outputs, including any set-up that may need to be performed with respect to global variables.

Values returned through the interface refer to values that are either passed through parameters or messages; values that the function call itself returns in the style of a "C" program function call; or values passed through global means (such as certain error routines in *ix-style operating systems).

In order to assure the description is complete, the evaluator uses other information provided about the module in the TOE design (e.g., algorithmic description, global data used) to ensure that it appears all data necessary for performing the functions of the module is presented to the module, and that any values that other modules expect the module under examination to provide are identified as being returned by the module. The evaluator determines accuracy by ensuring

that the description of the processing matches the information listed as being passed to or from an interface.

CC Part 3 ADV_TDS.5.8C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

13.8.5.4.14 Work unit ADV_TDS.5-13

The evaluator ***shall examine*** the TOE design to determine that it contains a complete and accurate mapping from the TSFI described in the functional specification to the modules of the TSF described in the TOE design.

The modules described in the TOE design provide a description of the implementation of the TSF. The TSFI provide a description of how the implementation is exercised. The evidence from the developer identifies the module that is initially invoked when an operation is requested at the TSFI, and identify the chain of modules invoked up to the module that is primarily responsible for implementing the functionality. However, a complete call tree for each TSFI is not required for this work unit. The cases in which more than one module would have to be identified are where there are "entry point" modules or wrapper modules that have no functionality other than conditioning inputs or de-multiplexing an input. Mapping to one of these modules would not provide any useful information to the evaluator.

The evaluator assesses the completeness of the mapping by ensuring that all of the TSFI map to at least one module. The verification of accuracy is more complex.

The first aspect of accuracy is that each TSFI is mapped to a module at the TSF boundary. This determination can be made by reviewing the module description and its interfaces/interactions. The next aspect of accuracy is that each TSFI identifies a chain of modules between the initial module identified and a module that is primarily responsible for implementing the function presented at the TSF. Note that this may be the initial module, or there may be several modules, depending on how much pre-conditioning of the inputs is done. It should be noted that one indicator of a pre-conditioning module is that it is invoked for a large number of the TSFI, where the TSFI are all of similar type (e.g., system call). The final aspect of accuracy is that the mapping makes sense. For instance, mapping a TSFI dealing with access control to a module that checks passwords is not accurate. The evaluator should again use judgement in making this determination. The goal is that this information aids the evaluator in understanding the system and implementation of the SFRs, and ways in which entities at the TSF boundary can interact with the TSF. The bulk of the assessment of whether the SFRs are described accurately by the modules is performed in other work units.

13.8.5.4.15 Work unit ADV_TDS.5-14

The evaluator ***shall examine*** the TOE security functional requirements and the TOE design, to determine that all ST security functional requirements are covered by the TOE design. The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems, and later to modules. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

For example, the FDP_ACC.1 Subset access control component contains an element with assignments. If the ST contained, for instance, ten rules in the FDP_ACC.1 Subset access control assignment, and these ten rules were implemented in specific places within fifteen modules, it would be inadequate for the evaluator to map FDP_ACC.1 Subset access control to one subsystem and claim the work unit had been completed. Instead, the evaluator would map FDP_ACC.1 Subset

access control (rule 1) to modules x, y and z of subsystem A; FDP_ACC.1 Subset access control (rule 2) to x, p, and q of subsystem A; etc.

13.8.5.4.16 Work unit ADV_TDS.5-15

The evaluator ***shall examine*** the TOE design to determine that it is an accurate instantiation of all security functional requirements.

The evaluator may construct a map between the TOE security functional requirements and the TOE design. This map will likely be from a functional requirement to a set of subsystems and modules. Note that this map may have to be at a level of detail below the component or even element level of the requirements, because of operations (assignments, refinements, selections) performed on the functional requirement by the ST author.

As an example, if the ST requirements specified a role-based access control mechanism, the evaluator would first identify the subsystems, and modules that contribute to this mechanism's implementation. This can be done by in-depth knowledge or understanding of the TOE design or by work done in the previous work unit. Note that this trace is only to identify the subsystems, and modules, and is not the complete analysis.

The next step would be to understand what mechanism the subsystems, and modules implemented. For instance, if the design described an implementation of access control based on UNIX-style protection bits, the design would not be an accurate instantiation of those access control requirements present in the ST example used above. If the evaluator cannot determine that the mechanism was accurately implemented because of a lack of detail, the evaluator would have to assess whether all of the SFR-enforcing subsystems and modules have been identified, or if adequate detail had been provided for those subsystems and modules.

13.8.6 Evaluation of sub-activity (ADV_TDS.6)

CC Part 3 ADV_TDS.6.1C: *The design shall describe the structure of the TOE in terms of subsystems.*

CC Part 3 ADV_TDS.6.2C: *The design shall describe the TSF in terms of modules, designating each module as SFR-enforcing, SFR-supporting, or SFR-non-interfering.*

CC Part 3 ADV_TDS.6.3C: *The design shall identify all subsystems of the TSF.*

CC Part 3 ADV_TDS.6.4C: *The design shall provide a semiformal description of each subsystem of the TSF, supported by informal, explanatory text where appropriate.*

CC Part 3 ADV_TDS.6.5C: *The design shall provide a description of the interactions among all subsystems of the TSF.*

CC Part 3 ADV_TDS.6.6C: *The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.*

CC Part 3 ADV_TDS.6.7C: *The design shall* ***describe*** *each module in* ***semiformal style in*** *terms of its purpose, interaction, interfaces, return values from those interfaces, and called interfaces to other modules, supported by informal, explanatory text where appropriate.*

CC Part 3 ADV_TDS.6.8C: *The formal specification of the TSF subsystems shall describe the TSF using a formal style, supported by informal, explanatory text where appropriate.*

CC Part 3 ADV_TDS.6.9C: *The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.*

CC Part 3 ADV_TDS.6.10C: *The proof of correspondence between the formal specifications of the TSF subsystems and of the functional specification shall demonstrate that all behaviour described in the TOE design is a correct and complete refinement of the TSFI that invoked it.*

There is no general guidance; the evaluation authority should be consulted for guidance on this sub-activity.

13.9 Composite design compliance (ADV_COMP)

13.9.1 General

The composite-specific work units defined here are intended to be integrated as refinements to the evaluation activities of the ADV class listed in the following table. The other activities of the ADV class do not require composite-specific work units.

Table 2 — ADV_COMP

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
ADV_ARC	ADV_ARC.1.1E	ADV_ARC.1-1	ADV_COMP.1-1
ADV_IMP	ADV_IMP.1.1E	ADV_IMP.1-1	ADV_COMP.1-1
ADV_TDS	ADV_TDS.1.2E	ADV_TDS.1-7	ADV_COMP.1-1

NOTE If the level of the assurance requirement chosen is higher than those identified in this table, the composite-specific work unit is also applicable.

13.9.2 Evaluation of sub-activity (ADV_COMP.1)

13.9.2.1 Objectives

The aim of this activity is to determine whether the requirements on the dependent component, imposed by the related base component, are fulfilled in the composite product.

13.9.2.2 Application notes

The requirements on the dependent component, imposed by the related base component, can be formulated in the relevant base component-related user guidance, *ETR for composite evaluation* (e.g. in form of observations and recommendations) and amendments to the base component evaluation as contained in the corresponding report of the base component evaluation authority (e.g. in form of further constraints, stipulations and recommendations). The developer of the dependent component shall regard each of these sources, if available, and implement the dependent component in such a way that the applicable requirements are appropriately implemented and accordingly fulfilled. The composite product evaluator shall verify that all the relevant requirements for the dependent component that are imposed by the base component and provided in its evaluation related documentation have been taken into account by the dependent component developer and are fulfilled by the composite product.

The composite product evaluation sponsor shall ensure that the following is made available for the composite product evaluator:

- the base component-related user guidance;
- the base component-related *ETR for composite evaluation* prepared by the base component evaluator;
- the base component's report of the base component evaluation authority;

- a rationale for secure composite product implementation including evidence prepared by the dependent component developer.

The TSF of the composite product are represented at various levels of abstraction in the families of the development class ADV. From experience, the appropriate levels of design representation for examining, whether the requirements of the base component are fulfilled by the composite product, are the TOE design (ADV_TDS), security architecture (ADV_ARC) and the implementation (ADV_IMP). In case that these design representation levels are not available (e.g. due to the assurance package chosen is EAL1), the current family is not applicable (see the next paragraph for the reason).

Due to the definition of the composite product the interface between its base component and dependent component is the internal one, hence, a functional specification (ADV_FSP) as representation level is not appropriate for analysing the design compliance.

Security architecture ADV_ARC as assurance family is dedicated to ensure that integrative security services like domain separation, self-protection and non-bypassability properly work. It is impossible and not the sense of the composite evaluation to have an insight into the architectural internals of the related base component (it is a matter of the base component evaluation). What the composite product evaluator shall do in the context of ADV_ARC is:

a) to determine whether the dependent component uses services of the related base component within its own composite product Security Target to provide domain separation, self-protection, non-bypassability and protected start-up; in the case of no, there are no further composite activities for ADV_ARC; in the case of yes, then:

b) the evaluator shall determine whether the dependent component uses these services of the base component in an appropriate/secure way (please refer to the base component user guidance and further sources for requirements).

As consistency of the composite product security policy has already been considered in the context of the Security Target in the assurance family ASE_COMP, there is no necessity to consider non-contradictoriness of the security policy model (ADV_SPM) of the composite product and the security policy model of its related base component.

13.9.2.3 Action ADV_COMP.1.1E

13.9.2.3.1 General

CC Part 3 ADV_COMP.1.1C: *The design compliance justification shall provide a rationale for design compliance -- on an appropriate representation level -- of how the requirements on the dependent component that are imposed by the related base component are fulfilled in the composite product.*

13.9.2.3.2 Work unit ADV_COMP.1-1

The evaluator ***shall examine*** the rationale for design compliance to determine that all applicable requirements on the dependent component that are imposed by the base component are fulfilled by the composite product.

In order to perform this work unit the evaluator shall use the rationale for design compliance as well as the TSF representation on the ADV_TDS, ADV_ARC and ADV_IMP levels on the one side and the input of the base component in form of the base component-related user guidance, *ETR for composite evaluation*, and report of the base component evaluation authority on the other side. The evaluator shall analyse which base component requirements are applicable for the current composite product, based on the identified RP_SFR-MECH and RP_SFR-SERV (refer to ASE_COMP). The evaluator shall compare each of the applicable requirements with the actual specification

and/or implementation of the composite product and determine, for each requirement, whether it is fulfilled. As a result, the evaluator confirms or disproves the rationale for design compliance.

For example, the base component guidance may require the dependent component to perform a special start-up sequence testing the current state of the base component and initialising its self-protection mechanisms. Such information can be found in the description of security architecture ADV_ARC of the composite product.

As a second example, the base component guidance may require the dependent component to perform a DFA check on the DES operation, while the dependent component is implementing BAC in an e-passport MRTD. The ADV_ARC will explain whether the base component guidance is followed up or not, and, in case that the requirements in the base component guidance are not followed, a corresponding reasoning will be provided. The arguments of the developer explain why a non-compliancy will not introduce vulnerabilities.

The appropriate representation level (ADV_TDS, ADV_ARC and/or ADV_IMP), on which the analysis is being performed, can be chosen and mixed flexibly depending on the concrete composite product and the requirement in question. Where it is not self-explaining, the evaluator shall justify why the representation level chosen is appropriate.

The evaluator activities in the context of this work unit can be spread over different single evaluation aspects (e.g. over ADV_TDS and ADV_IMP). In this case the evaluator performs the partial activity in the context of the corresponding single evaluation aspect. Then the notation for this work unit shall be ADV_COMP.1-1-TDS, ADV_COMP.1-1-ARC and ADV_COMP.1-1-IMP, respectively.

If the assurance package chosen does not contain the families ADV_TDS, ADV_ARC or ADV_IMP (e.g. EAL1), this work unit is not applicable.

The result of this work unit shall be integrated to the result of ADV_TDS.1.2E / ADV_TDS.1-7, ADV_ARC.1.1E / ADV_ARC.1-1, and ADV_IMP.1.1E / ADV_IMP.1-1 (or the equivalent higher components if a higher assurance level is selected).

14 Class AGD: Guidance documents

14.1 General

The purpose of the guidance document activity is to judge the adequacy of the documentation describing how the user can handle the TOE in a secure manner. Such documentation should take into account the various types of users (e.g. those who accept, install, administrate or operate the TOE) whose incorrect actions can adversely affect the security of the TOE or of their own data.

The guidance documents class is subdivided into two families which are concerned firstly with the preparative procedures (all that has to be done to transform the delivered TOE into its evaluated configuration in the environment as described in the ST, i.e. accepting and installing the TOE) and secondly with the operational user guidance (all that has to be done during the operation of the TOE in its evaluated configuration, i.e. operation and administration).

14.2 Application notes

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.

14.3 Operational user guidance (AGD_OPE)

14.3.1 Evaluation of sub-activity (AGD_OPE.1)

14.3.1.1 Objectives

The objectives of this sub-activity are to determine whether the user guidance describes for each user role the security functionality and interfaces provided by the TSF, provides instructions and guidelines for the secure use of the TOE, addresses secure procedures for all modes of operation, facilitates prevention and detection of insecure TOE states, or whether it is misleading or unreasonable.

14.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design, if applicable;

d) the user guidance.

14.3.1.3 Action AGD_OPE.1.1E

14.3.1.3.1 General

CC Part 3 AGD_OPE.1.1C: *The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.*

14.3.1.3.2 Work unit AGD_OPE.1-1

The evaluator ***shall examine*** the operational user guidance to determine that it describes, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

The configuration of the TOE may allow different user roles to have dissimilar privileges in making use of the different functions of the TOE. This means that some users are authorized to perform certain functions, while other users may not be so authorized. These functions and privileges should be described, for each user role, by the user guidance.

The user guidance identifies, for each user role, the functions and privileges that must be controlled, the types of commands required for them, and the reasons for such commands. The user guidance should contain warnings regarding the use of these functions and privileges. Warnings should address expected effects, possible side effects, and possible interactions with other functions and privileges.

CC Part 3 AGD_OPE.1.2C: *The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.*

14.3.1.3.3 Work unit AGD_OPE.1-2

The evaluator ***shall examine*** the operational user guidance to determine that it describes, for each user role, the secure use of the available interfaces provided by the TOE.

The user guidance should provide advice regarding effective use of the TSF (e.g. reviewing password composition practises, suggested frequency of user file backups, discussion on the effects of changing user access privileges).

CC Part 3 AGD_OPE.1.3C: *The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.*

14.3.1.3.4 Work unit AGD_OPE.1-3

The evaluator ***shall examine*** the operational user guidance to determine that it describes, for each user role, the available security functionality and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

The user guidance should contain an overview of the security functionality that is visible at the user interfaces.

The user guidance should identify and describe the purpose, behaviour, and interrelationships of the security interfaces and functionality.

For each user-accessible interface, 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, their particular purposes, valid and default values, and secure and insecure use settings of such parameters, both individually or in combination;

c) describe the immediate TSF response, message, or code returned.

The evaluator should consider the functional specification and the ST to determine that the TSF described in these documents is consistent to the operational user guidance. The evaluator has to ensure that the operational user guidance is complete to allow the secure use through the TSFI available to all types of human users. The evaluator may, as an aid, prepare an informal mapping between the guidance and these documents. Any omissions in this mapping may indicate incompleteness.

CC Part 3 AGD_OPE.1.4C: *The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.*

14.3.1.3.5 Work unit AGD_OPE.1-4

The evaluator ***shall examine*** the operational user guidance to determine that it describes, for each user role, each type of security-relevant event relative to the user functions that need to be performed, including changing the security characteristics of entities under the control of the TSF and operation following failure or operational error.

All types of security-relevant events are detailed for each user role, such that each user knows what events may occur and what action (if any) they 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 user intervention to maintain secure operation.

CC Part 3 AGD_OPE.1.5C: *The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.*

14.3.1.3.6 Work unit AGD_OPE.1-5

The evaluator ***shall examine*** the operational user 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.

Other evaluation evidence, particularly the functional specification, provide an information source that the evaluator should use to determine that the guidance contains sufficient guidance information.

If test documentation is included in the assurance package, then the information provided in this evidence can also be used to determine that the guidance contains sufficient guidance documentation. The detail provided in the test steps can be used to confirm that the guidance provided is sufficient for the use and administration of the TOE.

The evaluator should focus on a single human visible TSFI at a time, comparing the guidance for securely using the TSFI with other evaluation evidence, to determine that the guidance related to the TSFI is sufficient for the secure usage (i.e. consistent with the SFRs) of that TSFI. The evaluator should also consider the relationships between interfaces, searching for potential conflicts.

CC Part 3 AGD_OPE.1.6C: *The operational user guidance shall, for each user role, describe the security controls to be followed in order to fulfil the security objectives for the operational environment as described in the ST.*

14.3.1.3.7 Work unit AGD_OPE.1-6

The evaluator ***shall examine*** the operational user guidance to determine that it describes, for each user role, the security measures to be followed in order to fulfil the security objectives for the operational environment as described in the ST.

The evaluator analyses the security objectives for the operational environment in the ST and determines that for each user role, the relevant security measures are described appropriately in the user guidance.

The security measures described in the user guidance should include all relevant external procedural, physical, personnel and connectivity measures.

Note that those measures relevant for secure installation of the TOE are examined in Preparative procedures (AGD_PRE).

CC Part 3 AGD_OPE.1.7C: *The operational user guidance shall be clear and reasonable.*

14.3.1.3.8 Work unit AGD_OPE.1-7

The evaluator ***shall examine*** the operational user guidance to determine that it is clear.

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.

14.3.1.3.9 Work unit AGD_OPE.1-8

The evaluator ***shall examine*** the operational user guidance to determine that it is reasonable.

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.

14.4 Preparative procedures (AGD_PRE)

14.4.1 Evaluation of sub-activity (AGD_PRE.1)

14.4.1.1 Objectives

The objective of this sub-activity is to determine whether the procedures and steps for the secure preparation of the TOE have been documented and result in a secure configuration.

14.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE including its preparative procedures;

c) the description of developer's delivery procedures, if applicable;

14.4.1.3 Application notes

The preparative procedures refer to all acceptance and installation procedures, that are necessary to progress the TOE to the secure configuration as described in the ST.

14.4.1.4 Action AGD_PRE.1.1E

14.4.1.4.1 General

CC Part 3 AGD_PRE.1.1C: *The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.*

14.4.1.4.2 Work unit AGD_PRE.1-1

The evaluator ***shall examine*** the provided acceptance procedures to determine that they describe the steps necessary for secure acceptance of the TOE in accordance with the developer's delivery procedures.

If it is not anticipated by the developer's delivery procedures that acceptance procedures will or can be applied, this work unit is not applicable, and is therefore considered to be satisfied.

The acceptance procedures should include as a minimum, that the user has to check that all parts of the TOE as indicated in the ST have been delivered in the correct version.

The acceptance procedures should reflect the steps the user has to perform in order to accept the delivered TOE that are implied by the developer's delivery procedures.

The acceptance procedures should provide detailed information about the following, if applicable:

a) making sure that the delivered TOE is the complete evaluated instance;

b) detecting modification/masquerading of the delivered TOE.

CC Part 3 AGD_PRE.1.2C: *The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.*

14.4.1.4.3 Work unit AGD_PRE.1-2

The evaluator ***shall examine*** the provided installation procedures to determine that they describe the steps necessary for secure installation of the TOE and the secure preparation of the operational environment in accordance with the security objectives in the ST.

If it is not anticipated that installation procedures will or can be applied (e.g. because the TOE may already be delivered in an operational state), this work unit is not applicable, and is therefore considered to be satisfied.

The installation procedures should provide detailed information about the following, if applicable:

a) minimum system requirements for secure installation;

b) requirements for the operational environment in accordance with the security objectives provided by the ST;

c) the steps the user has to perform in order to get to an operational TOE being commensurate with its evaluated configuration. Such a description shall include - for each step - a clear scheme for the decision on the next step depended on success, failure or problems at the current step;

d) changing the installation specific security characteristics of entities under the control of the TSF (for example parameters, settings, passwords);

e) handling exceptions and problems.

14.4.1.5 Action AGD_PRE.1.2E

14.4.1.5.1 Work unit AGD_PRE.1-3

The evaluator ***shall perform*** all user procedures necessary to prepare the TOE to determine that the TOE and its operational environment can be prepared securely using only the supplied preparative procedures.

Preparation requires the evaluator to advance the TOE from a deliverable state to the state in which it is operational, including acceptance and installation of the TOE, and enforcing the SFRs consistent with the security objectives for the TOE specified in the ST.

The evaluator should follow only the developer's procedures and may perform the activities that customers are usually expected to perform to accept and install the TOE, using the supplied preparative procedures only. Any difficulties encountered during such an exercise may be indicative of incomplete, unclear or unreasonable guidance.

This work unit may be performed in conjunction with the evaluation activities under Independent testing (ATE_IND).

If it is known that the TOE will be used as a dependent component for a composed TOE evaluation, then the evaluator should ensure that the operational environment is satisfied by the base component used in the composed TOE.

15 Class ALC: Life-cycle support

15.1 General

The purpose of the life-cycle support activity is to determine the adequacy of the security procedures that the developer uses during the development and maintenance of the TOE. These procedures include the life-cycle model used by the developer, the configuration management, the security measures used throughout TOE development, the tools used by the developer throughout the life-cycle of the TOE, the handling of security flaws, and the delivery activity.

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. A measurable life-cycle model used for the TOE can remove ambiguity in assessing the development progress of the TOE.

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.

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.

The use of well-defined development tools and the application of implementation standards by the developer and by third parties involved in the development process help to ensure that vulnerabilities are not inadvertently introduced during refinement.

The flaw remediation activity is intended to track security flaws, to identify corrective actions, and to distribute the corrective action information to TOE users.

The purpose of the delivery activity is to judge the adequacy of the documentation of the procedures used to ensure that the TOE is delivered to the consumer without modification.

15.2 CM capabilities (ALC_CMC)

15.2.1 Evaluation of sub-activity (ALC_CMC.1)

15.2.1.1 Objectives

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

15.2.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing.

15.2.1.3 Action ALC_CMC.1.1E

15.2.1.3.1 General

CC Part 3 ALC_CMC.1.1C: *The TOE shall be labelled with its unique reference.*

15.2.1.3.2 Work unit ALC_CMC.1-1

The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This can 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).

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.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

15.2.1.3.3 Work unit ALC_CMC.1-2

The evaluator ***shall check*** that the TOE references used are consistent.

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 also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed IT TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the IT TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

15.2.2 Evaluation of sub-activity (ALC_CMC.2)

15.2.2.1 Objectives

The objectives of this sub-activity are to determine whether the developer uses a CM system that uniquely identifies all configuration items.

15.2.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

15.2.2.3 Application notes

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 Evaluation of sub-activity (ALC_CMC.3) the requirements are expanded beyond these two items, and more explicit evidence of operation is required.

15.2.2.4 Action ALC_CMC.2.1E

15.2.2.4.1 General

CC Part 3 ALC_CMC.2.1C: *The TOE shall be labelled with its unique reference.*

15.2.2.4.2 Work unit ALC_CMC.2-1

The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This can 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).

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.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

15.2.2.4.3 Work unit ALC_CMC.2-2

The evaluator ***shall check*** that the TOE references used are consistent.

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 also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed IT TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the IT TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

CC Part 3 ALC_CMC.2.2C: *The CM documentation shall describe the method used to uniquely identify the configuration items.*

15.2.2.4.4 Work unit ALC_CMC.2-3

The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

Procedures should describe how the status of each configuration item can be tracked throughout the life-cycle of the TOE. The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:

a) the method how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;

b) the method 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.

CC Part 3 ALC_CMC.2.3C: *The CM system shall uniquely identify all configuration items.*

15.2.2.4.5 Work unit ALC_CMC.2-4

The evaluator ***shall examine*** the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

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.

15.2.3 Evaluation of sub-activity (ALC_CMC.3)

15.2.3.1 Objectives

The objectives of this sub-activity are to determine whether the developer uses a CM system that uniquely identifies all configuration items, and whether the ability to modify these items is properly controlled.

15.2.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

15.2.3.3 Action ALC_CMC.3.1E

15.2.3.3.1 General

CC Part 3 ALC_CMC.3.1C: *The TOE shall be labelled with its unique reference.*

15.2.3.3.2 Work unit ALC_CMC.3-1

The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This can 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).

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.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

15.2.3.3.3 Work unit ALC_CMC.3-2

The evaluator ***shall check*** that the TOE references used are consistent.

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 also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed IT TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the IT TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

CC Part 3 ALC_CMC.3.2C: *The CM documentation shall describe the method used to uniquely identify the configuration items.*

15.2.3.3.4 Work unit ALC_CMC.3-3

The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

Procedures should describe how the status of each configuration item can be tracked throughout the life-cycle of the TOE. The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:

- the method how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;
- the method how configuration items are assigned unique identifiers and how they are entered into the CM system;
- the method to be used to identify superseded versions of a configuration item.

CC Part 3 ALC_CMC.3.3C: *The CM system shall uniquely identify all configuration items.*

15.2.3.3.5 Work unit ALC_CMC.3-4

The evaluator ***shall examine*** the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

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.

CC Part 3 ALC_CMC.3.4C: *The CM system shall provide controls such that only authorized changes are made to the configuration items.*

15.2.3.3.6 Work unit ALC_CMC.3-5

The evaluator ***shall examine*** the CM access control measures described in the CM plan to determine that they are effective in preventing unauthorized access to the configuration items.

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 cannot be bypassed. The evaluator may use the outputs generated by the CM system procedures required by ALC_CMC.3.8C. The evaluator may also witness a demonstration of the CM system to ensure that the access control measures employed are operating effectively.

CC Part 3 ALC_CMC.3.5C: *The CM documentation shall include a CM plan.*

15.2.3.3.7 Work unit ALC_CMC.3-6

The evaluator ***shall check*** that the CM documentation provided includes a CM plan.

The CM plan needs not to be a connected document, but it is recommended that there is a single document that describes where the various parts of the CM plan can be found. If the CM plan is no single document, the list in the following work unit gives hints regarding which context is expected.

CC Part 3 ALC_CMC.3.6C: *The CM plan shall describe how the CM system is used for the development of the TOE.*

15.2.3.3.8 Work unit ALC_CMC.3-7

The evaluator ***shall examine*** the CM plan to determine that it describes how the CM system is used for the development of the TOE.

The descriptions contained in a CM plan include, if applicable:

a) all activities performed in the TOE development that are subject to configuration management procedures (e.g. creation, modification or deletion of a configuration item, data-backup, archiving);

b) which means (e.g. CM tools, forms) have to be made available;

c) the usage of the CM tools: the necessary details for a user of the CM system to be able to operate the CM tools correctly in order to maintain the integrity of the TOE;

d) which other objects (development components, tools, assessment environments) are taken under CM control;

e) the roles and responsibilities of individuals required to perform operations on individual configuration items (different roles may be identified for different types of configuration items, e.g. design documentation or source code);

f) how CM instances (e.g. change control boards, interface control working groups) are introduced and staffed;

g) the description of the change management, including the process of verifying that the proposed change is necessary and the consequence would be acceptable;

h) the procedures that are used to ensure that only authorized individuals can make changes to configuration items;

i) the procedures that are used to ensure that concurrency problems do not occur as a result of simultaneous changes to configuration items;

j) 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 can 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 can be recorded in an audit trail of changes made or change control records;

k) 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).

CC Part 3 ALC_CMC.3.7C: *The evidence shall demonstrate that all configuration items are being maintained under the CM system.*

15.2.3.3.9 Work unit ALC_CMC.3-8

The evaluator ***shall check*** that the configuration items identified in the configuration list are being maintained by the CM system.

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. design documents 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 needs to 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.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.3.8C: *The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.*

15.2.3.3.10 Work unit ALC_CMC.3-9

The evaluator ***shall check*** the CM documentation to ascertain that it includes the CM system records identified by the CM plan.

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 ALC_CMC.3.7C. Example output can include change control forms, or configuration item access approval forms.

15.2.3.3.11 Work unit ALC_CMC.3-10

The evaluator ***shall examine*** the evidence to determine that the CM system is being operated in accordance with the CM plan.

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.

For guidance on sampling see A.2**,** Sampling.

Further confidence in the correct operation of the CM system and the effective maintenance of configuration items may be established by means of interviews with selected development staff. In conducting such interviews, the evaluator aims to gain a deeper understanding of how the CM system is used in practise 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.

It is expected that the evaluator will visit the development site in support of this activity.

For guidance on site visits see A.4**,** Site Visits.

15.2.4 Evaluation of sub-activity (ALC_CMC.4)

15.2.4.1 Objectives

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 by automated tools, thus making the CM system less susceptible to human error or negligence.

15.2.4.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

15.2.4.3 Action ALC_CMC.4.1E

15.2.4.3.1 General

CC Part 3 ALC_CMC.4.1C: *The TOE shall be labelled with its unique reference.*

15.2.4.3.2 Work unit ALC_CMC.4-1

The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This can 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).

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.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

15.2.4.3.3 Work unit ALC_CMC.4-2

The evaluator ***shall check*** that the TOE references used are consistent.

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 also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the composed TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

CC Part 3 ALC_CMC.4.2C: *The CM documentation shall describe the method or methods used to uniquely identify the configuration items.*

15.2.4.3.4 Work unit ALC_CMC.4-3

The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

Procedures should describe how the status of each configuration item can be tracked throughout the life-cycle of the TOE. The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:

a) the method how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;

b) the method 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.

CC Part 3 ALC_CMC.4.3C: *The CM system shall uniquely identify all configuration items.*

15.2.4.3.5 Work unit ALC_CMC.4-4

The evaluator ***shall examine*** the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

Assurance that the CM system uniquely identifies all configuration items is gained by examining the identifiers for the configuration items. For configuration items identified under ALC_CMS, 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.

CC Part 3 ALC_CMC.4.4C: *The CM system shall provide automated controls such that only authorized changes are made to the configuration items.*

15.2.4.3.6 Work unit ALC_CMC.4-5

The evaluator ***shall examine*** the CM access control measures described in the CM plan (cf. ALC_CMC.4.6C) to determine that they are automated and effective in preventing unauthorized access to the configuration items.

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 cannot be bypassed. The evaluator may use the outputs generated by the CM system procedures required by ALC_CMC.4.10C. The evaluator may also witness a demonstration of the CM system to ensure that the access control measures employed are operating effectively.

CC Part 3 ALC_CMC.4.5C: *The CM system shall support the production of the TOE by automated means.*

15.2.4.3.7 Work unit ALC_CMC.4-6

The evaluator ***shall check*** the CM plan (cf.ALC_CMC.4.6C) for automated procedures for supporting the production of the TOE.

The term "production" applies to those processes adopted by the developer to progress the TOE from the implementation representation to a state acceptable for delivery to the end customer.

The evaluator verifies the existence of automated production support procedures within the CM plan.

The following are examples for automated means supporting the production of the TOE:

a) a "make" tool (as provided with many software development tools) in the case of a software TOE;

b) a tool ensuring automatically (for example by means of bar codes) that only parts are combined which indeed belong together in the case of a hardware TOE.

15.2.4.3.8 Work unit ALC_CMC.4-7

The evaluator ***shall examine*** the TOE production support procedures to determine that they are effective in ensuring that a TOE is generated that reflects its implementation representation.

The production support procedures should describe which tools have to be used to produce the final TOE from the implementation representation in a clearly defined way. The conventions, directives, or other necessary constructs are described under ALC_TAT.

The evaluator determines that by following the production support procedures the correct configuration items would be used to generate the TOE. For example, in a software TOE this may include checking that the automated production procedures ensure that all source files and

related libraries are included in the compiled object code. Moreover, the procedures should ensure that compiler options and comparable other options are defined uniquely. For a hardware TOE, this work unit may include checking that the automatic production procedures ensure that the belonging parts are built together and no parts are missing.

The customer can then be confident that the version of the TOE delivered for installation is derived from the implementation representation in an unambiguous way and implements the SFRs as described in the ST.

The evaluator should bear in mind that the CM system need not necessarily possess the capability to produce the TOE, but should provide support for the process that will help reduce the probability of human error.

CC Part 3 ALC_CMC.4.6C: *The CM documentation shall include a CM plan.*

15.2.4.3.9 Work unit ALC_CMC.4-8

The evaluator ***shall check*** that the CM documentation provided includes a CM plan.

The CM plan does not need to be contained within a single document, but it is recommended that there is a separate document that describes where the various parts of the CM plan can be found. If the CM plan is provided by a set of documents, the list in the following work unit gives guidance regarding the required content.

CC Part 3 ALC_CMC.4.7C: *The CM plan shall describe how the CM system is used for the development of the TOE.*

15.2.4.3.10 Work unit ALC_CMC.4-9

The evaluator ***shall examine*** the CM plan to determine that it describes how the CM system is used for the development of the TOE.

The descriptions contained in a CM plan include, if applicable:

a) all activities performed in the TOE development that are subject to configuration management procedures (e.g. creation, modification or deletion of a configuration item, data-backup, archiving);

b) which means (e.g. CM tools, forms) have to be made available;

c) the usage of the CM tools: the necessary details for a user of the CM system to be able to operate the CM tools correctly in order to maintain the integrity of the TOE;

d) the production support procedures;

e) which other objects (development components, tools, assessment environments) are taken under CM control;

f) the roles and responsibilities of individuals required to perform operations on individual configuration items (different roles may be identified for different types of configuration items, e.g. design documentation or source code);

g) how CM instances (e.g. change control boards, interface control working groups) are introduced and staffed;

h) the description of the change management;

i) the procedures that are used to ensure that only authorized individuals can make changes to configuration items;

j) the procedures that are used to ensure that concurrency problems do not occur as a result of simultaneous changes to configuration items;

k) 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 can 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 can be recorded in an audit trail of changes made or change control records;

l) 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).

CC Part 3 ALC_CMC.4.8C: *The CM plan shall describe the procedures used to accept modified or newly created configuration items as part of the TOE.*

15.2.4.3.11 Work unit ALC_CMC.4-10

The evaluator ***shall examine*** the CM plan to determine that it describes the procedures used to accept modified or newly created configuration items as parts of the TOE.

The descriptions of the acceptance procedures in the CM plan should include the developer roles or individuals responsible for the acceptance and the criteria to be used for acceptance. In order to meet the desired assurance level, the acceptance criteria may include a suite of tests that determines whether the required security objective and/or performance objective is met. The criteria should take into account all acceptance situations that may occur, in particular:

a) accepting an item into the CM system for the first time, in particular inclusion of software, firmware and hardware components from other manufacturers into the TOE ("integration");

b) moving configuration items to the next life-cycle phase at each stage of the construction of the TOE (e.g. module, subsystem, system);

c) subsequent to transports between different development sites.

If this work unit is applied to a dependent component that is going to be integrated in a composed TOE, the CM plan should consider the control of base components obtained by the dependent TOE developer.

When obtaining the components the evaluators are to verify the following:

- Transfer of each base component from the base component developer to the integrator (dependent TOE developer) was performed in accordance with the base component TOE's secure delivery procedures, as reported in the base component TOE certification report.
- The component received has the same identifiers as those stated in the ST and Certification Report for the component TOE.
- All additional material required by a developer for composition (integration) is provided. This is to include the necessary extract of the component TOE's functional specification.

CC Part 3 ALC_CMC.4.9C: *The evidence shall demonstrate that all configuration items are being maintained under the CM system.*

15.2.4.3.12 Work unit ALC_CMC.4-11

The evaluator ***shall check*** that the configuration items identified in the configuration list are being maintained by the CM system.

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. design documents 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 needs to 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.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.4.10C: *The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.*

15.2.4.3.13 Work unit ALC_CMC.4-12

The evaluator ***shall check*** the CM documentation to ascertain that it includes the CM system records identified by the CM plan.

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 ALC_CMC.4.9C. Example output can include change control forms, or configuration item access approval forms.

15.2.4.3.14 Work unit ALC_CMC.4-13

The evaluator ***shall examine*** the evidence to determine that the CM system is being operated in accordance with the CM plan.

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.

For guidance on sampling see A.2**,** Sampling.

Further confidence in the correct operation of the CM system and the effective maintenance of configuration items may be established by means of interviews with selected development staff. In conducting such interviews, the evaluator aims to gain a deeper understanding of how the CM system is used in practise 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.

It is expected that the evaluator will visit the development site in support of this activity.

For guidance on site visits see A.4**,** Site Visits.

15.2.5 Evaluation of sub-activity (ALC_CMC.5)

15.2.5.1 Objectives

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 by automated tools, thus making the CM system less susceptible to human error or negligence.

15.2.5.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the TOE suitable for testing;

c) the configuration management documentation.

15.2.5.3 Action ALC_CMC.5.1E

15.2.5.3.1 General

CC Part 3 ALC_CMC.5.1C: *The TOE shall be labelled with its unique reference.*

15.2.5.3.2 Work unit ALC_CMC.5-1

The evaluator ***shall check*** that the TOE provided for evaluation is labelled with its reference.

The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This can 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).

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.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

15.2.5.3.3 Work unit ALC_CMC.5-2

The evaluator ***shall check*** that the TOE references used are consistent.

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 also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed IT TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the IT TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If

the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

CC Part 3 ALC_CMC.5.2C: *The CM documentation shall describe the method used to uniquely identify the configuration items.*

15.2.5.3.4 Work unit ALC_CMC.5-3

The evaluator ***shall examine*** the method of identifying configuration items to determine that it describes how configuration items are uniquely identified.

Procedures should describe how the status of each configuration item can be tracked throughout the life-cycle of the TOE. The procedures may be detailed in the CM plan or throughout the CM documentation. The information included should describe:

a) the method how each configuration item is uniquely identified, such that it is possible to track versions of the same configuration item;

b) the method 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.

CC Part 3 ALC_CMC.5.3C: *The CM documentation shall justify that the acceptance procedures provide for an adequate and appropriate review of changes to all configuration items.*

15.2.5.3.5 Work unit ALC_CMC.5-4

The evaluator ***shall examine*** the CM documentation to determine that it justifies that the acceptance procedures provide for an adequate and appropriate review of changes to all configuration items.

The CM documentation should make it sufficiently clear that by following the acceptance procedures only parts of adequate quality are incorporated into the TOE.

CC Part 3 ALC_CMC.5.4C: *The CM system shall uniquely identify all configuration items.*

15.2.5.3.6 Work unit ALC_CMC.5-5

The evaluator ***shall examine*** the configuration items to determine that they are identified in a way that is consistent with the CM documentation.

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.

CC Part 3 ALC_CMC.5.5C: *The CM system shall provide automated controls such that only authorized changes are made to the configuration items.*

15.2.5.3.7 Work unit ALC_CMC.5-6

The evaluator ***shall examine*** the CM access control measures described in the CM plan (cf. ALC_CMC.5.12C) to determine that they are automated and effective in preventing unauthorized access to the configuration items.

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 cannot be bypassed. The evaluator may use the outputs generated by the CM system procedures required by ALC_CMC.5.16C. The evaluator may also witness a demonstration of the CM system to ensure that the access control measures employed are operating effectively.

CC Part 3 ALC_CMC.5.6C: *The CM system shall support the production of the TOE by automated means.*

15.2.5.3.8 Work unit ALC_CMC.5-7

The evaluator ***shall check*** the CM plan (cf. ALC_CMC.5.12C) for automated procedures for supporting the production of the TOE.

The term "production" applies to those processes adopted by the developer to progress the TOE from the implementation representation to a state acceptable for delivery to the end customer.

The evaluator verifies the existence of automated production support procedures within the CM plan.

The following are examples for automated means supporting the production of the TOE:

a) a "make" tool (as provided with many software development tools) in the case of a software TOE;

b) a tool ensuring automatically (for example by means of bar codes) that only parts are combined which indeed belong together in the case of a hardware TOE.

15.2.5.3.9 Work unit ALC_CMC.5-8

The evaluator ***shall examine*** the TOE production support procedures to determine that they are effective in ensuring that a TOE is generated that reflects its implementation representation.

The production support procedures should describe which tools have to be used to produce the final TOE from the implementation representation in a clearly defined way. The conventions, directives, or other necessary constructs are described under ALC_TAT.

The evaluator determines that by following the production support procedures the correct configuration items would be used to generate the TOE. For example, in a software TOE this may include checking that the automated production procedures ensure that all source files and related libraries are included in the compiled object code. Moreover, the procedures should ensure that compiler options and comparable other options are defined uniquely. For a hardware TOE, this work unit may include checking that the automatic production procedures ensure that the belonging parts are built together and no parts are missing.

The customer can then be confident that the version of the TOE delivered for installation is derived from the implementation representation in an unambiguous way and implements the SFRs as described in the ST.

The evaluator should bear in mind that the CM system need not necessarily possess the capability to produce the TOE, but should provide support for the process that will help reduce the probability of human error.

CC Part 3 ALC_CMC.5.7C: *The CM system shall ensure that the person responsible for accepting a configuration item into CM is not the person who developed it.*

15.2.5.3.10 Work unit ALC_CMC.5-9

The evaluator ***shall examine*** the CM system to determine that it ensures that the person responsible for accepting a configuration item is not the person who developed it.

The acceptance procedures describe who is responsible for accepting a configuration item. From these descriptions, the evaluator should be able to determine that the person who developed a configuration item is in no case responsible for its acceptance.

CC Part 3 ALC_CMC.5.8C: *The CM system shall identify the configuration items that comprise the TSF.*

15.2.5.3.11 Work unit ALC_CMC.5-10

The evaluator ***shall examine*** the CM system to determine that it identifies the configuration items that comprise the TSF.

The CM documentation should describe how the CM system identifies the configuration items that comprise the TSF. The evaluator should select a sample of configuration items covering each type of items, particularly containing TSF and non-TSF items, and check that they are correctly classified by the CM system.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.5.9C: *The CM system shall support the audit of all changes to the TOE by automated means, including the originator, date, and time in the audit trail.*

15.2.5.3.12 Work unit ALC_CMC.5-11

The evaluator ***shall examine*** the CM system to determine that it supports the audit of all changes to the TOE by automated means, including the originator, date, and time in the audit trail.

The evaluator should inspect a sample of audit trails and check, if they contain the minimum information.

CC Part 3 ALC_CMC.5.10C: *The CM system shall provide an automated means to identify all other configuration items that are affected by the change of a given configuration item.*

15.2.5.3.13 Work unit ALC_CMC.5-12

The evaluator ***shall examine*** the CM system to determine that it provides an automated means to identify all other configuration items that are affected by the change of a given configuration item.

The CM documentation should describe how the CM system identifies all other configuration items that are affected by the change of a given configuration item. The evaluator should select a sample of configuration items, covering all types of items, and exercise the automated means to determine that it identifies all items that are affected by the change of the selected item.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.5.11C: *The CM system shall be able to identify the version of the implementation representation from which the TOE is generated.*

15.2.5.3.14 Work unit ALC_CMC.5-13

The evaluator ***shall examine*** the CM system to determine that it is able to identify the version of the implementation representation from which the TOE is generated.

The CM documentation should describe how the CM system identifies the version of the implementation representation from which the TOE is generated. The evaluator should select a sample of the parts used to produce the TOE and should apply the CM system to verify that it identifies the corresponding implementation representation in the correct version.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.5.12C: *The CM documentation shall include a CM plan.*

15.2.5.3.15 Work unit ALC_CMC.5-14

The evaluator ***shall check*** that the CM documentation provided includes a CM plan.

The CM plan needs not to be a connected document, but it is recommended that there is a single document that describes where the various parts of the CM plan can be found. If the CM plan is no single document, the list in the following work unit gives hints regarding which context is expected.

CC Part 3 ALC_CMC.5.13C: *The CM plan shall describe how the CM system is used for the development of the TOE.*

15.2.5.3.16 Work unit ALC_CMC.5-15

The evaluator ***shall examine*** the CM plan to determine that it describes how the CM system is used for the development of the TOE.

The descriptions contained in a CM plan include, if applicable:

a) all activities performed in the TOE development that are subject to configuration management procedures (e.g. creation, modification or deletion of a configuration item, data-backup, archiving);

b) which means (e.g. CM tools, forms) have to be made available;

c) the usage of the CM tools: the necessary details for a user of the CM system to be able to operate the CM tools correctly in order to maintain the integrity of the TOE;

d) the production support procedures;

e) which other objects (development components, tools, assessment environments) are taken under CM control;

f) the roles and responsibilities of individuals required to perform operations on individual configuration items (different roles may be identified for different types of configuration items, e.g. design documentation or source code);

g) how CM instances (e.g. change control boards, interface control working groups) are introduced and staffed;

h) the description of the change management;

i) the procedures that are used to ensure that only authorized individuals can make changes to configuration items;

j) the procedures that are used to ensure that concurrency problems do not occur as a result of simultaneous changes to configuration items;

k) 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 can 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 can be recorded in an audit trail of changes made or change control records;

l) 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).

CC Part 3 ALC_CMC.5.14C: *The CM plan shall describe the procedures used to accept modified or newly created configuration items as part of the TOE.*

15.2.5.3.17 Work unit ALC_CMC.5-16

The evaluator ***shall examine*** the CM plan to determine that it describes the procedures used to accept modified or newly created configuration items as parts of the TOE.

The descriptions of the acceptance procedures in the CM plan should include the developer roles or individuals responsible for the acceptance and the criteria to be used for acceptance. In order to meet the desired assurance level, the acceptance criteria may include a suite of tests that determines whether the required security objective and/or performance objective is met. The criteria should take into account all acceptance situations that may occur, in particular:

a) accepting an item into the CM system for the first time, in particular inclusion of software, firmware and hardware components from other manufacturers into the TOE ("integration");

b) moving configuration items to the next life-cycle phase at each stage of the construction of the TOE (e.g. module, subsystem, system);

c) subsequent to transports between different development sites.

If this work unit is applied to a dependent component that is going to be integrated in a composed TOE, the CM plan should consider the control of base components obtained by the dependent TOE developer.

When obtaining the components the evaluators are to verify the following:

Transfer of each base component from the base component developer to the integrator (dependent TOE developer) was performed in accordance with the base component TOE's secure delivery procedures, as reported in the base component TOE certification report.

The component received has the same identifiers as those stated in the ST and Certification Report for the component TOE.

All additional material required by a developer for composition (integration) is provided. This is to include the necessary extract of the component TOE's functional specification.

CC Part 3 ALC_CMC.5.15C: *The evidence shall demonstrate that all configuration items are being maintained under the CM system.*

15.2.5.3.18 Work unit ALC_CMC.5-17

The evaluator ***shall check*** that the configuration items identified in the configuration list are being maintained by the CM system.

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. design documents 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 needs to 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.

For guidance on sampling see A.2**,** Sampling.

CC Part 3 ALC_CMC.5.16C: *The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.*

15.2.5.3.19 Work unit ALC_CMC.5-18

The evaluator ***shall check*** the CM documentation to ascertain that it includes the CM system records identified by the CM plan.

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 ALC_CMC.5.15C. Example output can include change control forms, or configuration item access approval forms.

15.2.5.3.20 Work unit ALC_CMC.5-19

The evaluator ***shall examine*** the evidence to determine that the CM system is being operated in accordance with the CM plan.

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.

For guidance on sampling see A.2**,** Sampling.

Further confidence in the correct operation of the CM system and the effective maintenance of configuration items may be established by means of interviews with selected development staff. In conducting such interviews, the evaluator aims to gain a deeper understanding of how the CM system is used in practise 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.

It is expected that the evaluator will visit the development site in support of this activity.

For guidance on site visits see A.4**,** Site Visits.

15.2.5.4 Action ALC_CMC.5.2E

15.2.5.4.1 Work unit ALC_CMC.5-20

The evaluator ***shall examine*** the production support procedures to determine that by following these procedures a TOE would be produced like that one provided by the developer for testing activities.

If the TOE is a small software TOE and production consists of compiling and linking, the evaluator can confirm the adequacy of the production support procedures by reapplying them.

If the production process of the TOE is more complicated (as for example in the case of a smart card), but has already started, the evaluator should inspect the application of the production support procedures during a visit of the development site. They can compare a copy of the TOE produced in their presence with the samples used for their testing activities.

For guidance on site visits see A.4**,** Site Visits.

Otherwise the evaluator's determination should be based on the documentary evidence provided by the developer.

This work unit may be performed in conjunction with the evaluation activities under Implementation representation (ADV_IMP)**.**

15.3 CM scope (ALC_CMS)

15.3.1 Evaluation of sub-activity (ALC_CMS.1)

15.3.1.1 Objectives

The objective of this sub-activity is to determine whether the developer performs configuration management on the TOE and the evaluation evidence. These configuration items are controlled in accordance with CM capabilities (ALC_CMC)**.**

15.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the configuration list.

15.3.1.3 Action ALC_CMS.1.1E

15.3.1.3.1 General

CC Part 3 ALC_CMS.1.1C: *The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.*

15.3.1.3.2 Work unit ALC_CMS.1-1

The evaluator ***shall check*** that the configuration list includes the following set of items:

a) the TOE itself;

b) the evaluation evidence required by the SARs in the ST.

CC Part 3 ALC_CMS.1.2C: *The configuration list shall uniquely identify the configuration items.*

15.3.1.3.3 Work unit ALC_CMS.1-2

The evaluator ***shall examine*** the configuration list to determine that it uniquely identifies each configuration item.

The configuration list contains 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.

15.3.2 Evaluation of sub-activity (ALC_CMS.2)

15.3.2.1 Objectives

The objective of this sub-activity is to determine whether the configuration list includes the TOE, the parts that comprise the TOE, and the evaluation evidence. These configuration items are controlled in accordance with CM capabilities (ALC_CMC)**.**

15.3.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the configuration list.

15.3.2.3 Action ALC_CMS.2.1E

15.3.2.3.1 General

CC Part 3 ALC_CMS.2.1C: *The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; and the parts that comprise the TOE.*

15.3.2.3.2 Work unit ALC_CMS.2-1

The evaluator ***shall check*** that the configuration list includes the following set of items:

a) the TOE itself;

b) the parts that comprise the TOE;

c) the evaluation evidence required by the SARs.

CC Part 3 ALC_CMS.2.2C: *The configuration list shall uniquely identify the configuration items.*

15.3.2.3.3 Work unit ALC_CMS.2-2

The evaluator ***shall examine*** the configuration list to determine that it uniquely identifies each configuration item.

The configuration list contains 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.

CC Part 3 ALC_CMS.2.3C: *For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.*

15.3.2.3.4 Work unit ALC_CMS.2-3

The evaluator ***shall check*** that the configuration list indicates the developer of each TSF relevant configuration item.

If only one developer is involved in the development of the TOE, this work unit is not applicable, and is therefore considered to be satisfied.

15.3.3 Evaluation of sub-activity (ALC_CMS.3)

15.3.3.1 Objectives

The objective of this sub-activity is to determine whether the configuration list includes the TOE, the parts that comprise the TOE, the TOE implementation representation, and the evaluation evidence. These configuration items are controlled in accordance with CM capabilities (ALC_CMC).

15.3.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the configuration list.

15.3.3.3 Action ALC_CMS.3.1E

15.3.3.3.1 General

CC Part 3 ALC_CMS.3.1C: *The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; the parts that comprise the TOE; and the implementation representation.*

15.3.3.3.2 Work unit ALC_CMS.3-1

The evaluator ***shall check*** that the configuration list includes the following set of items:

a) the TOE itself;

b) the parts that comprise the TOE;

c) the TOE implementation representation;

d) the evaluation evidence required by the SARs in the ST.

CC Part 3 ALC_CMS.3.2C: *The configuration list shall uniquely identify the configuration items.*

15.3.3.3.3 Work unit ALC_CMS.3-2

The evaluator ***shall examine*** the configuration list to determine that it uniquely identifies each configuration item.

The configuration list contains 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.

CC Part 3 ALC_CMS.3.3C: *For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.*

15.3.3.3.4 Work unit ALC_CMS.3-3

The evaluator ***shall check*** that the configuration list indicates the developer of each TSF relevant configuration item.

If only one developer is involved in the development of the TOE, this work unit is not applicable, and is therefore considered to be satisfied.

15.3.4 Evaluation of sub-activity (ALC_CMS.4)

15.3.4.1 Objectives

The objective of this sub-activity is to determine whether the configuration list includes the TOE, the parts that comprise the TOE, the TOE implementation representation, security flaws, and the evaluation evidence. These configuration items are controlled in accordance with CM capabilities (ALC_CMC)**.**

15.3.4.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the configuration list.

15.3.4.3 Action ALC_CMS.4.1E

15.3.4.3.1 General

CC Part 3 ALC_CMS.4.1C: *The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; the parts that comprise the TOE; the implementation representation; and security flaw reports and resolution status.*

15.3.4.3.2 Work unit ALC_CMS.4-1

The evaluator ***shall check*** that the configuration list includes the following set of items:

a) the TOE itself;

b) the parts that comprise the TOE;

c) the TOE implementation representation;

d) the evaluation evidence required by the SARs in the ST;

e) 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 database).

CC Part 3 ALC_CMS.4.2C: *The configuration list shall uniquely identify the configuration items.*

15.3.4.3.3 Work unit ALC_CMS.4-2

The evaluator ***shall examine*** the configuration list to determine that it uniquely identifies each configuration item.

The configuration list contains 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.

CC Part 3 ALC_CMS.4.3C: *For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.*

15.3.4.3.4 Work unit ALC_CMS.4-3

The evaluator ***shall check*** that the configuration list indicates the developer of each TSF relevant configuration item.

If only one developer is involved in the development of the TOE, this work unit is not applicable, and is therefore considered to be satisfied.

15.3.5 Evaluation of sub-activity (ALC_CMS.5)

15.3.5.1 Objectives

The objective of this sub-activity is to determine whether the configuration list includes the TOE, the parts that comprise the TOE, the TOE implementation representation, security flaws, development tools and related information, and the evaluation evidence. These configuration items are controlled in accordance with CM capabilities (ALC_CMC).

15.3.5.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the configuration list.

15.3.5.3 Action ALC_CMS.5.1E

15.3.5.3.1 General

CC Part 3 ALC_CMS.5.1C: *The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; the parts that comprise the TOE; the implementation representation; security flaw reports and resolution status; and development tools and related information.*

15.3.5.3.2 Work unit ALC_CMS.5-1

The evaluator ***shall check*** that the configuration list includes the following set of items:

a) the TOE itself;

b) the parts that comprise the TOE;

c) the TOE implementation representation;

d) the evaluation evidence required by the SARs in the ST;

e) 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 database);

f) all tools (incl. test software, if applicable) involved in the development and production of the TOE including the names, versions, configurations and roles of each development tool, and related documentation.

For a software TOE, "development tools" are usually programming languages and compiler and "related documentation" comprises compiler and linker options. For a hardware TOE, "development tools" can be hardware design languages, simulation and synthesis tools, compilers, and "related documentation" can comprise compiler options again.

CC Part 3 ALC_CMS.5.2C: *The configuration list shall uniquely identify the configuration items.*

15.3.5.3.3 Work unit ALC_CMS.5-2

The evaluator ***shall examine*** the configuration list to determine that it uniquely identifies each configuration item.

The configuration list contains 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.

CC Part 3 ALC_CMS.5.3C: *For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.*

15.3.5.3.4 Work unit ALC_CMS.5-3

The evaluator ***shall check*** that the configuration list indicates the developer of each TSF relevant configuration item.

If only one developer is involved in the development of the TOE, this work unit is not applicable, and is therefore considered to be satisfied.

15.4 Delivery (ALC_DEL)

15.4.1 Evaluation of sub-activity (ALC_DEL.1)

15.4.1.1 Objectives

The objective of this sub-activity is to determine whether the delivery documentation describes all procedures used to maintain security of the TOE when distributing the TOE to the user.

15.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the delivery documentation.

15.4.1.3 Action ALC_DEL.1.1E

15.4.1.3.1 General

CC Part 3 ALC_DEL.1.1C: *The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to the consumer.*

15.4.1.3.2 Work unit ALC_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 consumer.

The delivery documentation describes proper procedures to maintain security of the TOE during transfer of the TOE or its component parts and to determine the identification of the TOE.

The delivery documentation should cover the entire TOE, but may contain different procedures for different parts of the TOE. The evaluation should consider the totality of procedures.

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). Standard commercial practise for packaging and delivery may be acceptable. This includes shrink wrapped packaging, a security tape or a sealed envelope. For the distribution, physical (e.g. public mail or a private distribution service) or electronic (e.g. electronic mail or downloading off the Internet) procedures may be used.

Cryptographic checksums or a software signature may be used by the developer to ensure that tampering or masquerading can be detected. Tamper proof seals additionally indicate if the confidentiality has been broken. For software TOEs, confidentiality can be assured by using encryption. If availability is of concern, a secure transportation might be required.

Interpretation of the term "necessary to maintain security" will need to consider:

- The nature of the TOE (e.g. whether it is software or hardware).
- The overall security level stated for the TOE by the chosen level of the Vulnerability Assessment. If the TOE is required to be resistant against attackers of a certain potential in its intended environment, this should also apply to the delivery of the TOE. The evaluator should determine that a balanced approach has been taken, such that delivery does not present a weak point in an otherwise secure development process.
- The security objectives provided by the ST. The emphasis in the delivery documentation is likely to be on measures related to integrity, as integrity of the TOE is always important. 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.

15.4.1.4 Implied evaluator action

CC Part 3 ALC_DEL.1.2D: *The developer shall use the delivery procedures*

15.4.1.4.1 Work unit ALC_DEL.1-2

The evaluator ***shall examine*** aspects of the delivery process to determine that the delivery procedures are used.

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 seeks some assurance that they are applied in practise. 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 after the user has received it (e.g. checking for tamper proof seals);

c) observing that the process is applied in practise when the evaluator obtains the TOE through regular channels;

d) questioning end users as to how the TOE was delivered.

For guidance on site visits see A.4**,** Site Visits.

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.

15.5 Development security (ALC_DVS)

15.5.1 Evaluation of sub-activity (ALC_DVS.1)

15.5.1.1 Objectives

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.

15.5.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the development security documentation.

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 Evaluation of sub-activity (ALC_CMC.4) "Production support and acceptance procedures" and the Evaluation of sub-activity (ALC_CMS.4) "Problem tracking CM coverage"). Evidence that the procedures are being applied is also required.

15.5.1.3 Action ALC_DVS.1.1E

15.5.1.3.1 General

CC Part 3 ALC_DVS.1.1C: *The development security documentation shall describe all the physical, logical, procedural, personnel, and other security controls that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.*

15.5.1.3.2 Work unit 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.

The evaluator determines what is necessary by first referring to the ST for any information that may assist in the determination of necessary protection.

If no explicit information is available from the ST the evaluator will need to make a determination of the necessary measures. 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.

The following types of security measures are considered by the evaluator when examining the documentation:

- physical, for example physical access controls used to prevent unauthorised access to the TOE development environment (during normal working hours and at other times);
- 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 within and out of the development environment and between different development sites in accordance with defined acceptance procedures,
- 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.

- personnel, for example any controls or checks made to establish the trustworthiness of new development staff;
- other security measures, for example the logical protections on any development machines.

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 and for transports between different locations. For example, development can occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites. Transports of parts of the TOE or the unfinished TOE between different development sites are to be covered by Development security (ALC_DVS), whereas the transport of the finished TOE to the consumer is dealt with in Delivery (ALC_DEL).

Development includes the production of the TOE.

15.5.1.3.3 Work unit 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.

The evaluator should examine whether the following is included in the policies:

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 unauthorized modification in order to preserve the integrity of the TOE, and which members of the development staff are allowed to modify such material.

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.

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 CM capabilities (ALC_CMC). 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.

Whereas the CM capabilities (ALC_CMC) requirements are fixed, those for the Development security (ALC_DVS), mandating only necessary measures, are dependent on the nature of the TOE, and on information that may be provided in the ST. The evaluators would then determine that such a policy had been applied under this sub-activity.

15.5.1.4 Action ALC_DVS.1.2E

15.5.1.4.1 Work unit ALC_DVS.1-3

The evaluator ***shall examine*** the development security documentation and associated evidence to determine that the security measures are being applied.

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 can 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.

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 evaluation authority.

For guidance on site visits see A.4, Site Visits.

15.5.2 Evaluation of sub-activity (ALC_DVS.2)

15.5.2.1 Objectives

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. Additionally, sufficiency of the measures as applied is intended be justified.

15.5.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the development security documentation.

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 Evaluation of sub-activity (ALC_CMC.4) "Production support and acceptance procedures" and the Evaluation of sub-activity (ALC_CMS.4) "Problem tracking CM coverage"). Evidence that the procedures are being applied is also required.

15.5.2.3 Action ALC_DVS.2.1E

15.5.2.3.1 General

CC Part 3 ALC_DVS.2.1C: *The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment*

15.5.2.3.2 Work unit ALC_DVS.2-1

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.

The evaluator determines what is necessary by first referring to the ST for any information that may assist in the determination of necessary protection.

If no explicit information is available from the ST the evaluator will need to make a determination of the necessary measures. 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.

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 unauthorized access to the TOE development environment (during normal working hours and at other times);

b) procedural, for example covering:

   i. granting of access to the development environment or to specific parts of the environment such as development machines;

   ii. revocation of access rights when a person leaves the development team;

   iii. transfer of protected material out of the development environment and between different development sites in accordance with defined acceptance procedures;

   iv. admitting and escorting visitors to the development environment.

c) roles and responsibilities in ensuring the continued application of security measures, and the detection of security breaches;

d) personnel, for example any controls or checks made to establish the trustworthiness of new development staff;

e) other security measures, for example the logical protections on any development machines.

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 and for transports between different locations. For example, development can occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites. Transports of parts of the TOE or the unfinished TOE between different development sites are to be covered by the Development security (ALC_DVS), whereas the transport of the finished TOE to the consumer is dealt with in the Delivery (ALC_DEL).

Development includes the production of the TOE.

CC Part 3 ALC_DVS.2.2C: *The development security documentation shall justify that the security controls provide the necessary level of protection to maintain the confidentiality and integrity of the TOE.*

15.5.2.3.3 Work unit ALC_DVS.2-2

The evaluator ***shall examine*** the development security documentation to determine that an appropriate justification is given why the security measures provide the necessary level of protection to maintain the confidentiality and integrity of the TOE.

Since attacks on the TOE or its related information are assumed in different design and production stages, measures and procedures need to have an appropriate level necessary to prevent those attacks or to make them more difficult.

Since this level depends on the overall attack potential claimed for the TOE (cf. the Vulnerability analysis (AVA_VAN) component chosen), the development security documentation should justify the necessary level of protection to maintain the confidentiality and integrity of the TOE. This level has to be achieved by the security measures applied.

The concept of protection measures should be consistent, and the justification should include an analysis of how the measures are mutually supportive. All aspects of development and production on all the different sites with all roles involved up to delivery of the TOE should be analysed.

Justification may include an analysis of potential vulnerabilities taking the applied security measures into account.

There may be a convincing argument showing that e.g.:

- The technical measures and mechanisms of the developer's infrastructure are sufficient for keeping the appropriate security level (e.g. cryptographic mechanisms as well as physical protection mechanisms, properties of the CM system (cf. ALC_CMC.4-5)];
- The system containing the implementation representation of the TOE (including concerning guidance documents) provides effective protection against logical attacks e.g. by "Trojan" code or viruses. It can be adequate, if the implementation representation is kept on an isolated system where only the software necessary to maintain it is installed and where no additional software is installed afterwards.
- Data brought into this system need to be carefully considered to prevent the installation of hidden functionality onto the system. The effectiveness of these measures need to be tested, e.g. by independently trying to get access to the machine, install some additional executable or get some information out of the machine using logical attacks.
- The appropriate organisational (procedural and personal) measures are unconditionally enforced.

15.5.2.3.4 Work unit ALC_DVS.2-3

The evaluator ***shall examine*** the development confidentiality and integrity policies in order to determine the sufficiency of the security measures employed.

The evaluator should examine whether the following is included in the policies:

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 unauthorized modification in order to preserve the integrity of the TOE, and which members of the development staff are allowed to modify such material.

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.

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 CM capabilities (ALC_CMC). 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.

Whereas the CM capabilities (ALC_CMC) requirements are fixed, those for the Development security (ALC_DVS), mandating only necessary measures, are dependent on the nature of the TOE, and on information that may be provided in the ST. For example, the ST may identify a security objective for the development environment that requires the TOE to be developed by staff that has security clearance. The evaluators would then determine that such a policy had been applied under this sub-activity.

15.5.2.4 Action ALC_DVS.2.2E

15.5.2.4.1 Work unit ALC_DVS.2-4

The evaluator ***shall examine*** the development security documentation and associated evidence to determine that the security measures are being applied.

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 can 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.

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 evaluation authority.

For guidance on site visits see A.4**,** Site Visits.

15.6 Flaw remediation (ALC_FLR)

15.6.1 Evaluation of sub-activity (ALC_FLR.1)

15.6.1.1 Objectives

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.

15.6.1.2 Input

The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation.

15.6.1.3 Action ALC_FLR.1.1E

15.6.1.3.1 General

CC Part 3 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.*

15.6.1.3.2 Work unit 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.

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 time frame, from initial detection through ascertaining that the flaw is a security flaw, to resolution of the security flaw.

If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the Flaw remediation (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.

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.

CC Part 3 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.*

15.6.1.3.3 Work unit 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.

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 can be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password "BACK DOOR".

15.6.1.3.4 Work unit 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.

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.

CC Part 3 ALC_FLR.1.3C: *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

15.6.1.3.5 Work unit 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.

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).

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.

CC Part 3 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 TOE users.*

15.6.1.3.6 Work unit 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.

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.

TOE users may be provided with 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.

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 can 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.

15.6.2 Evaluation of sub-activity (ALC_FLR.2)

15.6.2.1 Objectives

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.

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

15.6.2.2 Input

The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation;

b) law remediation guidance documentation.

15.6.2.3 Action ALC_FLR.2.1E

15.6.2.3.1 General

CC Part 3 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.*

15.6.2.3.2 Work unit 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.

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 time frame, from initial detection through ascertaining that the flaw is a security flaw, to resolution of the security flaw.

If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the Flaw remediation (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.

CC Part 3 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*

15.6.2.3.3 Work unit 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.

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 can be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password "BACKDOOR".

15.6.2.3.4 Work unit 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.

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.

CC Part 3 ALC_FLR.2.3C: *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

15.6.2.3.5 Work unit 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.

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).

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.

CC Part 3 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 TOE users.*

15.6.2.3.6 Work unit 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.

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.

TOE users may be provided with 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.

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 can 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.

CC Part 3 ALC_FLR.2.5C: *The flaw remediation procedures shall describe a means by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.*

15.6.2.3.7 Work unit 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.

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.

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 can 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.

CC Part 3 ALC_FLR.2.6C: *The procedures for processing reported security flaws shall ensure that any reported flaws are remediated and the remediation procedures issued to TOE users.*

15.6.2.3.8 Work unit 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.

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.

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.

15.6.2.3.9 Work unit 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 remediation procedures for each security flaw.

The procedures describe the process that is taken from the point at which a security flaw is resolved to the point at which the remediation procedures are 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 ALC_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 be expected to be distributed by bonded courier. Updates unrelated to flaw remediation would follow the procedures set forth in the documentation meeting the Delivery (ALC_DEL) requirements.

CC Part 3 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.*

15.6.2.3.10 Work unit 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.

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.

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.

CC Part 3 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.*

15.6.2.3.11 Work unit 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.

15.6.3 Evaluation of sub-activity (ALC_FLR.3)

15.6.3.1 Objectives

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.

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.

15.6.3.2 Input

The evaluation evidence for this sub-activity is:

a) the flaw remediation procedures documentation;

b) flaw remediation guidance documentation.

15.6.3.3 Action ALC_FLR.3.1E

15.6.3.3.1 General

CC Part 3 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.*

15.6.3.3.2 Work unit 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.

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 time frame, from initial detection through ascertaining that the flaw is a security flaw, to resolution of the security flaw.

If a flaw is discovered not to be security-relevant, there is no need (for the purposes of the Flaw remediation (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.

CC Part 3 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.*

15.6.3.3.3 Work unit 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.

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 can be found that affects the identification and authentication enforced by the TSF by permitting authentication with the password "BACKDOOR".

15.6.3.3.4 Work unit 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.

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.

CC Part 3 ALC_FLR.3.3C: *The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.*

15.6.3.3.5 Work unit 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.

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).

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.

CC Part 3 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 TOE users.*

15.6.3.3.6 Work unit 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.

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.

TOE users may be provided with 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.

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 can 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.

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.

CC Part 3 ALC_FLR.3.5C: *The flaw remediation procedures shall describe a means by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.*

15.6.3.3.7 Work unit ALC_FLR.3-6

The evaluator ***shall examine*** the flaw remediation procedures to determine that the application of these procedures would result in a means for the developer to receive from TOE user reports of suspected security flaws or requests for corrections to such flaws.

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.

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 can 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.

CC Part 3 ALC_FLR.3.6C: *The flaw remediation procedures shall include a procedure requiring timely response and the automatic distribution of security flaw reports and the associated corrections to registered users who might be affected by the security flaw.*

15.6.3.3.8 Work unit ALC_FLR.3-7

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 can be affected with reports about, and associated corrections to, each security flaw.

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.

It is unnecessary to restrict the recipients of the reports and associated corrections to only those TOE users who can 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.

15.6.3.3.9 Work unit ALC_FLR.3-8

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 can be affected.

Automatic distribution does not mean that human interaction with the distribution method is not permitted. In fact, the distribution method can consist entirely of manual procedures, perhaps through a closely monitored procedure with prescribed escalation upon the lack of issue of reports or corrections.

It is unnecessary to restrict the recipients of the reports and associated corrections to only those TOE users who can 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.

CC Part 3 ALC_FLR.3.7C: *The procedures for processing reported security flaws shall ensure that any reported flaws are remediated and the remediation procedures issued to TOE users.*

15.6.3.3.10 Work unit ALC_FLR.3-9

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.

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 remediated. The procedures contain reasonable steps that show progress leading to the eventual, inevitable resolution.

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.

15.6.3.3.11 Work unit ALC_FLR.3-10

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 remediation procedures for each security flaw.

The procedures describe the process that is taken from the point at which a security flaw is resolved to the point at which the remediation procedures are provided. The procedures for delivering remediation procedures 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 Delivery (ALC_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 be expected to be distributed by bonded courier. Updates unrelated to flaw remediation would follow the procedures set forth in the documentation meeting the Delivery (ALC_DEL) requirements.

CC Part 3 ALC_FLR.3.8C: *The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.*

15.6.3.3.12 Work unit ALC_FLR.3-11

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.

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.

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.

CC Part 3 ALC_FLR.3.9C: *The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.*

15.6.3.3.13 Work unit ALC_FLR.3-12

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.

CC Part 3 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.*

15.6.3.3.14 Work unit 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.

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 can 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.

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.

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.

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).

CC Part 3 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.*

15.6.3.3.15 Work unit ALC_FLR.3-14

The evaluator ***shall examine*** the flaw remediation guidance to determine that it identifies specific points of contact for user reports and enquiries about security issues involving the TOE.

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.

15.7 Life-cycle definition (ALC_LCD)

15.7.1 Evaluation of sub-activity (ALC_LCD.1)

15.7.1.1 Objectives

The objective of this sub-activity is to determine whether the developer has used a documented model of the TOE life-cycle.

15.7.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the life-cycle definition documentation.

15.7.1.3 Action ALC_LCD.1.1E

15.7.1.3.1 General

CC Part 3 ALC_LCD.1.1C: *The life-cycle definition documentation shall describe the processes used to develop and maintain the TOE.*

15.7.1.3.2 Work unit 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.

The description of the life-cycle model should include:

a) information on the life-cycle phases of the TOE and the boundaries between the subsequent phases;

b) information on the procedures, tools and techniques used by the developer (e.g. for design, coding, testing, bug-fixing);

c) 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);

d) information on which parts of the TOE are delivered by subcontractors, if subcontractors are involved.

NOTE Evaluation of this sub-activity does not require the model used to conform to any standard life-cycle model.

CC Part 3 ALC_LCD.1.2C: *The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.*

15.7.1.3.3 Work unit 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.

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 covered under the CM capabilities (ALC_CMC)). Identified deficiencies in the life-cycle model will be of concern if they can reasonably be expected to give rise to the introduction of flaws into the TOE, either accidentally or deliberately.

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.

15.7.2 Evaluation of sub-activity (ALC_LCD.2)

15.7.2.1 Objectives

The objective of this sub-activity is to determine whether the developer has used a documented and measurable model of the TOE life-cycle.

15.7.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the life-cycle definition documentation;

c) information about the standard used;

d) the life-cycle output documentation.

15.7.2.3 Action ALC_LCD.2.1E

15.7.2.3.1 General

CC Part 3 ALC_LCD.2.1C: *The life-cycle definition documentation shall describe the model used to develop and maintain the TOE including the details of its arithmetic parameters and/or metrics used to measure the quality of the TOE and/or its development.*

15.7.2.3.2 Work unit ALC_LCD.2-1

The evaluator ***shall examine*** the documented description of the life-cycle model used to determine that it covers the development and maintenance process, including the details of its arithmetic parameters and/or metrics used to measure the TOE development.

The description of the life-cycle model includes:

a) information on the life-cycle phases of the TOE and the boundaries between the subsequent phases;

b) information on the procedures, tools and techniques used by the developer (e.g. for design, coding, testing, bug-fixing);

c) 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);

d) information on which parts of the TOE are delivered by subcontractors, if subcontractors are involved;

e) information on the parameters/metrics that are used to measure the TOE development. Metrics standards typically include guides for measuring and producing reliable products and cover the aspects reliability, quality, performance, complexity and cost. For the evaluation all those metrics are of relevance, which are used to increase quality by decreasing the probability of faults and thereby in turn increase assurance in the security of the TOE.

CC Part 3 ALC_LCD.2.2C: *The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.*

15.7.2.3.3 Work unit ALC_LCD.2-2

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.

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 covered under the CM capabilities (ALC_CMC)**).** Identified deficiencies in the life-cycle model will be of concern if they can reasonably be expected to give rise to the introduction of flaws into the TOE, either accidentally or deliberately.

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.

For the metrics/measurements used in the life-cycle model, evidence has to be provided that shows how those metrics/measurements usefully contribute to the minimisation of the likelihood of flaws. This can be viewed as the overall goal for measurement in an ALC context. As a consequence, the metrics/measurements have to be selected based on their capability to achieve that overall goal or contribute to that. In the first place a metric/measure is suitable with respect to ALC if a correlation between the metric/measure and the number of flaws can be stated with a certain degree of reliability. But also, a metric/measure useful for management purposes as for planning and monitoring the TOE development are helpful since badly managed projects are endangered to produce bad quality and to introduce flaws.

It may be possible to use metrics for quality improvement, for which this use is not obvious. For example, a metric to estimate the expected cost of a product development may help quality, if the developer can show that this is used to provide an adequate budget for development projects and that this helps to avoid quality problems arising from resource shortages.

It is not required that every single step in the life cycle of the TOE is measurable. However, the evaluator should see from the description of the measures and procedures that the metrics are appropriate to control the overall quality of the TOE and to minimise possible security flaws by this.

CC Part 3 ALC_LCD.2.3C: *The life-cycle output documentation shall provide the results of the measurements of the TOE development using the measurable life-cycle model.*

15.7.2.3.4 Work unit ALC_LCD.2-3

The evaluator ***shall examine*** the life-cycle output documentation to determine that it provides the results of the measurements of the TOE development using the measurable life-cycle model.

The results of the measurements and the life-cycle progress of the TOE should be in accordance with the life-cycle model.

The output documentation not only includes numeric values of the metrics but also documents actions taken as a result of the measurements and in accordance with the model. For example,

there may be a requirement that a certain design phase needs to be repeated, if some error rates measured during testing are outside of a defined threshold. In this case the documentation should show that such action was taken, if indeed the thresholds were not met.

If the evaluation is conducted in parallel with the development of the TOE it may be possible that quality measurements have not been used in the past. In this case the evaluator should use the documentation of the planned procedures in order to gain confidence that corrective actions are defined if results of quality measurements deviate from some threshold.

15.8 TOE Development Artifacts (ALC_TDA)

15.8.1 Evaluation of sub-activity (ALC_TDA.1)

15.8.1.1 Objectives

The objectives of this sub-activity are to determine whether the developer has recorded the unique identifiers of the implementation representation which has been used to generate the TOE.

15.8.1.1.1 Input

The evaluation evidence for this sub-activity is

a) the ST;

b) the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D;

c) the list of TOE implementation representation element names;

d) the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D;

e) the (author) origination information of the list of TOE implementation representation identifiers;

f) the developer documentation describing the following as required in CC Part 3 ALC_TDA.1.6D;

g) the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

h) the developer's timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

i) the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

j) the maintenance of the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information;

k) the developer's mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

l) the user manual of the developer's development tool which use the TOE implementation representation.

15.8.1.2 Action ALC_TDA.1.1E

15.8.1.2.1 General

CC Part 3 ALC_TDA.1.1C: *The list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall demonstrate the correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names.*

15.8.1.2.2 Work unit ALC_TDA.1-1

The evaluator ***shall examine*** the developer documentation describing the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to determine that there is a correspondence between the TOE implementation representation identifiers and the TOE implementation representation element names.

If the developer simply uses the unique TOE implementation representation identifiers as the TOE implementation representation element names, then the correspondence is trivial. Otherwise, the correspondence should have the following effect. If two elements within the TOE implementation representation share the same name, then either they are the same or they are separately identified by two distinct identifiers.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the TOE implementation representation element names are just file names. In that case, it is possible that those two files in the hard drive or in the cloud share the same name, but they have different contents. As a result, two distinct identifiers are necessary to distinguish them apart. The correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names therefore maps or links the two distinct identifiers to the same file name.

15.8.1.3 Action ALC_TDA.1.2E

15.8.1.3.1 General

CC Part 3 ALC_TDA.1.2C: *The TOE implementation representation element names shall be in the same form as used or referenced by the development tool to generate the TOE.*

15.8.1.3.2 Work unit ALC_TDA.1-2

The evaluator ***shall examine*** the user manual of the developer's development tool used to generate the TOE to determine that the development tool accepts the TOE implementation representation element names as its input parameters.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the evaluator only need to discover from the development tool user manual that the development tool accepts local or remote file names as its input parameters.

15.8.1.4 Action ALC_TDA.1.3E

15.8.1.4.1 General

CC Part 3 ALC_TDA.1.3C: *The timestamp of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the creation time of the TOE.*

15.8.1.4.2 Work unit ALC_TDA.1-3

The evaluator ***shall check*** the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D that it is consistent with the creation time of the TOE as referenced in the ST.

Consistency is confirmed by determining that the timestamp on the list of TOE implementation representation identifiers is earlier than the TOE creation time as referenced in the ST, and consistent with the time interval expected from the developer's build process (e.g. as described in deliverables for ALC_LCD).

15.8.1.5 Action ALC_TDA.1.4E

15.8.1.5.1 General

CC Part 3 ALC_TDA.1.4C: *The (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the (author) origination information of the TOE. The author origination information may be the name of an affiliate of an organization.*

15.8.1.5.2 Work unit ALC_TDA.1-4

The evaluator ***shall check*** the (author) origination information of the list of unique TOE implementation representation identifiers that it is consistent with the (author) origination information of the TOE as referenced in the ST.

This consistence means that the (author) origination of the list of unique TOE implementation representation identifiers is related to the (author) origination of the TOE in a reasonable manner.

EXAMPLE

A reasonable relationship is that the (author) origination of the list of unique TOE implementation representation identifiers is an employee, a contractor, a supplier, a subsidiary, or an organizational division of, or is identical to the (author) origination of the TOE.

15.8.1.6 Action ALC_TDA.1.5E

15.8.1.6.1 Work unit ALC_TDA.1-5

The evaluator ***shall check*** the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information by examining the developer documentation describing the maintenance of this integrity characteristic.

It is necessary that the developer documentation explains an applicable scenario where none of the following items is freely modified without any proper authorization after its initial existence:

a) the list of unique TOE implementation representation identifiers;

b) its associated timestamp;

c) its associated (author) origination information.

EXAMPLE

Applicable scenarios include the following:

a) these items reside in an access-controlled location and the associated access log/record indicates that these items have not been changed since their initial creation;

b) these items are written in a read only media;

c) these items are digitally signed and are verifiable with a valid public key.

15.8.1.7 Action ALC_TDA.1.6E

15.8.1.7.1 Work unit ALC_TDA.1-6

The evaluator ***shall examine*** the developer documentation describing the developer's mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to confirm the developer's ability to trace from the TOE to the list of unique TOE implementation representation identifiers.

It is necessary that the evaluator follows the developer documentation to find a list of identifiers using the TOE as its input and the evaluator checks that this list of identifiers matches the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D.

EXAMPLE

It is acceptable that the developer uses a tracing mechanism that directly or indirectly writes the list of TOE implementation representation identifiers or its representation to the TOE. Alternatively, it is also acceptable that the developer uses a bill of materials database as a tracing mechanism to maintain the correspondence between the TOE and the list of TOE implementation representation identifiers.

15.8.2 Evaluation of sub-activity (ALC_TDA.2)

15.8.2.1 Objectives

The objectives of this sub-activity are to determine whether the elements of implementation representation maintained under the configuration scope of ALC_CMS.3 are identifiable using the developer's unique identifiers of the implementation representation as recorded during the TOE generation time.

15.8.2.2 Input

The evaluation evidence for this sub-activity is

a) the ST;

b) the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D or ALC_TDA.2.1D;

c) the list of TOE implementation representation element names;

d) the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D or ALC_TDA.2.2D;

e) the (author) origination information of the list of TOE implementation representation identifiers;

f) the developer documentation describing the following as required in CC Part 3 ALC_TDA.1.6D or ALC_TDA.2.6D

g) the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

h) the developer's timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

i) the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

j) the maintenance of the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information;

k) the developer's mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

l) the user manual of the developer's development tool which use the TOE implementation representation;

m) the element names of implementation representation (as parts of the configuration list) under the configuration scope of CC Part 3 ALC_CMS.3.

15.8.2.3 Action ALC_TDA.2.1E

15.8.2.3.1 General

CC Part 3 ALC_TDA.2.1C: *The list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall demonstrate the correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names.*

15.8.2.3.2 Work unit ALC_TDA.2-1

The evaluator ***shall examine*** the developer documentation describing the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to determine that there is a correspondence between the TOE implementation representation identifiers and the TOE implementation representation element names.

If the developer simply uses the unique TOE implementation representation identifiers as the TOE implementation representation element names, then the correspondence is trivial. Otherwise, the correspondence should have the following effect. If two elements within the TOE implementation representation share the same name, then either they are the same or they are separately identified by two distinct identifiers.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the TOE implementation representation element names are just file names. In that case, it is possible that those two files in the hard drive or in the cloud share the same name, but they have different contents. As a result, two distinct identifiers are necessary to distinguish them apart. The correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names therefore maps or links the two distinct identifiers to the same file name.

15.8.2.4 Action ALC_TDA.2.2E

15.8.2.4.1 General

CC Part 3 ALC_TDA.2.2C: *The TOE implementation representation element names shall be in the same form as used or referenced by the development tool to generate the TOE.*

15.8.2.4.2 Work unit ALC_TDA.2-2

The evaluator ***shall examine*** the user manual of the developer's development tool used to generate the TOE to determine that the development tool accepts the TOE implementation representation element names as its input parameters.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the evaluator only need to discover from the development tool user manual that the development tool accepts local or remote file names as its input parameters.

15.8.2.5 Action ALC_TDA.2.3E

15.8.2.5.1 General

CC Part 3 ALC_TDA.2.3C: *The timestamp of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the creation time of the TOE.*

15.8.2.5.2 Work unit ALC_TDA.2-3

The evaluator ***shall check*** the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D that it is consistent with the creation time of the TOE as referenced in the ST.

Consistency is confirmed by determining that the timestamp on the list of TOE implementation representation identifiers is earlier than the TOE creation time as referenced in the ST, and consistent with the time interval expected from the developer's build process (e.g. as described in deliverables for ALC_LCD).

15.8.2.6 Action ALC_TDA.2.4E

15.8.2.6.1 General

CC Part 3 ALC_TDA.2.4C: *The (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the (author) origination information of the TOE. The author origination information may be the name of an affiliate of an organization.*

15.8.2.6.2 Work unit ALC_TDA.2-4

The evaluator ***shall check*** the (author) origination information of the list of unique TOE implementation representation identifiers that it is consistent with the (author) origination information of the TOE as referenced in the ST.

This consistence means that the (author) origination of the list of unique TOE implementation representation identifiers is related to the (author) origination of the TOE in a justifiable manner.

EXAMPLE

A reasonable relationship is that the (author) origination of the list of unique TOE implementation representation identifiers is an employee, a contractor, a supplier, a subsidiary, or an organizational division of, or is identical to the (author) origination of the TOE.

15.8.2.7 Action ALC_TDA.2.5E

15.8.2.7.1 General

CC Part 3 ALC_TDA.2.5C: *The list of identifiers of the elements of implementation representation under the configuration scope of ALC_CMS.3 shall match with the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.*

15.8.2.7.2 Work unit ALC_TDA.2-5

The evaluator ***shall check*** the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information by examining the developer documentation describing the maintenance of this integrity characteristic.

It is necessary that the developer documentation explains how unauthorized modification of any of the following items is prevented:

a) the list of unique TOE implementation representation identifiers;

b) its associated timestamp;

c) its associated (author) origination information.

Possible approaches include the following:

a) these items reside in an access-controlled location and the associated access log/record indicates that these items have not been changed since their initial creation;

b) these items are written in a read only media;

c) these items are digitally signed and are verifiable with a valid public key.

15.8.2.8 Action ALC_TDA.2.6E

15.8.2.8.1 Work unit ALC_TDA.2-6

The evaluator ***shall examine*** the developer documentation describing the developer's mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to confirm the developer's ability to trace from the TOE to the list of unique TOE implementation representation identifiers.

It is necessary that the evaluator follows the developer documentation to find a list of identifiers using the TOE as its input and the evaluator checks that this list of identifiers matches the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D.

EXAMPLE

It is acceptable that the developer uses a tracing mechanism that directly or indirectly writes the list of TOE implementation representation identifiers or its representation to the TOE. Alternatively, it is also acceptable that the developer uses a bill of materials database as a tracing mechanism to maintain the correspondence between the TOE and the list of TOE implementation representation identifiers.

15.8.2.9 Action ALC_TDA.2.7E

15.8.2.9.1 Work unit ALC_TDA.2-7

The evaluator ***shall check*** that the TOE implementation representation identifiers in the correspondence as determined in Work unit ALC_TDA.1-1 are capable to identify the element names of implementation representation (as parts of the configuration list) under the configuration scope of ALC_CMS.3.

15.8.3 Evaluation of sub-activity (ALC_TDA.3)

15.8.3.1 Objectives

A regenerated TOE copy is a TOE regeneration from another copy of the TOE implementation representation according to the developer's unique identifiers of the implementation representation as recorded during the TOE generation time. The objectives of this sub-activity are to determine that the developer's explanation of the functional differences, if any, between the regenerated TOE copy and the original TOE takes into account all visible differences, if any, between the regenerated TOE copy and the original TOE.

15.8.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D or ALC_TDA.2.1D or ALC_TDA.3.1D;

c) the list of TOE implementation representation element names;

d) the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D or ALC_TDA.2.2D or ALC_TDA.3.2D;

e) the (author) origination information of the list of TOE implementation representation identifiers;

f) the developer documentation describing the following as required in CC Part 3 ALC_TDA.1.6D or ALC_TDA.2.6D or ALC_TDA.3.6D;

g) the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

h) the developer’s timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

i) the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

j) the maintenance of the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information;

k) the developer’s mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;

l) the user manual of the developer's development tool which use the TOE implementation representation;

m) the element names of implementation representation (as parts of the configuration list) under the configuration scope of CC Part 3 ALC_CMS.3;

n) the inputs for the evaluation of ADV_IMP.1 sub-activity, which are:

   i. the implementation representation;

   ii. the documentation of the development tools, as resulting from ALC_TAT;

   iii. TOE design description.

o) the developer's explanation of the functional differences, if any, between the regenerated TOE copy and the original TOE as output from the developer action of CC Part 3 ALC_TDA.3.9D.

15.8.3.3 Action ALC_TDA.3.1E

15.8.3.3.1 General

CC Part 3 ALC_TDA.3.1C: *The list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall demonstrate the correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names.*

15.8.3.3.2 Work unit ALC_TDA.3-1

The evaluator ***shall examine*** the developer documentation describing the developer's creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to determine that there is a correspondence between the TOE implementation representation identifiers and the TOE implementation representation element names.

If the developer simply uses the unique TOE implementation representation identifiers as the TOE implementation representation element names, then the correspondence is trivial. Otherwise, the correspondence should have the following effect. If two elements within the TOE implementation representation share the same name, then either they are the same or they are separately identified by two distinct identifiers.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the TOE implementation representation element names are just file names. In that case, it is possible that those two files in the hard drive or in the cloud share the same name, but they have different contents. As a result, two distinct identifiers are necessary to distinguish them apart. The correspondence between the TOE implementation representation element identifiers and the TOE implementation representation element names therefore maps or links the two distinct identifiers to the same file name.

CC Part 3 ALC_TDA.3.2C: *The TOE implementation representation element names shall be in the same form as used or referenced by the development tool to generate the TOE.*

15.8.3.3.3 Work unit ALC_TDA.3-2

The evaluator ***shall examine*** the user manual of the developer's development tool used to generate the TOE to determine that the development tool accepts the TOE implementation representation element names as its input parameters.

EXAMPLE

If the TOE implementation representation elements are data files residing in a repository such as a hard drive or in the cloud, then the evaluator only need to discover from the development tool user manual that the development tool accepts local or remote file names as its input parameters.

CC Part 3 ALC_TDA.3.3C: *The timestamp of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the creation time of the TOE.*

15.8.3.3.4 Work unit ALC_TDA.3-3

The evaluator ***shall check*** the timestamp of the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.2D that it is consistent with the creation time of the TOE as referenced in the ST.

Consistency is confirmed by determining that the timestamp on the list of TOE implementation representation identifiers is earlier than the TOE creation time as referenced in the ST, and consistent with the time interval expected from the developer's build process (e.g. as described in deliverables for ALC_LCD).

15.8.3.4 Action ALC_TDA.3.4E

15.8.3.4.1 General

CC Part 3 ALC_TDA.3.4C: *The (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time shall be consistent with the (author) origination information of the TOE. The author origination information may be the name of an affiliate of an organization.*

15.8.3.4.2 Work unit ALC_TDA.3-4

The evaluator ***shall check*** the (author) origination information of the list of unique TOE implementation representation identifiers that it is consistent with the (author) origination information of the TOE as referenced in the ST.

This consistency means that the (author) origination of the list of unique TOE implementation representation identifiers is related to the (author) origination of the TOE professionally.

EXAMPLE

A reasonable relationship is that the (author) origination of the list of unique TOE implementation representation identifiers is an employee, a contractor, a supplier, a subsidiary, or an organizational division of, or is identical to the (author) origination of the TOE.

15.8.3.5 Action ALC_TDA.3.5E

15.8.3.5.1 General

CC Part 3 ALC_TDA.3.5C: *The list of identifiers of the elements of implementation representation under the configuration scope of ALC_CMS.3 shall match with the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.*

15.8.3.5.2 Work unit ALC_TDA.3-5

The evaluator ***shall check*** the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time and its associated timestamp and (author) origination information by examining the developer documentation describing the maintenance of this integrity characteristic.

It is necessary that the developer documentation explains an applicable scenario where none of the following items is freely modified without any proper authorization after its initial existence:

a) the list of unique TOE implementation representation identifiers;

b) its associated timestamp;

c) its associated (author) origination information.

EXAMPLE

Applicable scenarios include the following:

a) these items reside in an access-controlled location and the associated access log/record indicates that these items have not been changed since their initial creation;

b) these items are written in a read only media;

c) these items are digitally signed and are verifiable with a valid public key.

15.8.3.6 Action ALC_TDA.3.6E

15.8.3.6.1 General

CC Part 3 ALC_TDA.3.6C: *The developer's explanation of the functional differences, if any, between the regenerated TOE copy and the original TOE shall take into account all visible differences, if any, between the regenerated TOE copy and the original TOE.*

15.8.3.6.2 Work unit ALC_TDA.3-6

The evaluator ***shall examine*** the developer documentation describing the developer's mechanism to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time to confirm the developer's ability to trace from the TOE to the list of unique TOE implementation representation identifiers.

It is necessary that the evaluator follows the developer documentation to find a list of identifiers using the TOE as its input and the evaluator checks that this list of identifiers matches the list of TOE implementation representation identifiers as output from the developer action of CC Part 3 ALC_TDA.1.1D.

EXAMPLE

It is acceptable that the developer uses a tracing mechanism that directly or indirectly writes the list of TOE implementation representation identifiers or its representation to the TOE. Alternatively, it is also acceptable that the developer uses a bill of materials database as a tracing mechanism to maintain the correspondence between the TOE and the list of TOE implementation representation identifiers.

15.8.3.7 Action ALC_TDA.3.7E

15.8.3.7.1 Work unit ALC_TDA.3-7

The evaluator ***shall check*** that the TOE implementation representation identifiers in the correspondence as determined in Work unit ALC_TDA.1-1 are capable to identify the element names of implementation representation (as parts of the configuration list) under the configuration scope of ALC_CMS.3.

15.8.3.8 Action ALC_TDA.3.8E

15.8.3.8.1 Work unit ALC_TDA.3-8

The evaluator ***shall check*** that the developer's explanation of the functional differences, if any, between the regenerated TOE copy and the original TOE takes into account all visible differences, if any, between the regenerated TOE copy and the original TOE.

If parts of the TOE consist of software such as a collection of binary executable files, then it is possible to observe the corresponding differences between the regenerated TOE copy and the original TOE using a binary file editor/viewer or other applicable software diagnostic or testing tools.

If parts of the TOE consist of hardware such as integrated circuits, then it is possible to observe the corresponding differences between the regenerated TOE copy and the original TOE using a microscope or other applicable hardware diagnostic or testing tools.

In either case, if there is no visible difference between the regenerated TOE copy and the original TOE, then there is no functional difference for the developer to explain.

15.9 Tools and techniques (ALC_TAT)

15.9.1 Evaluation of sub-activity (ALC_TAT.1)

15.9.1.1 Objectives

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.

15.9.1.2 Input

The evaluation evidence for this sub-activity is:

a) the development tool documentation;

b) the subset of the implementation representation.

15.9.1.3 Application notes

This work may be performed in parallel with the evaluation activities under Implementation representation (ADV_IMP), specifically with regard to determining the use of features in the tools that will affect the object code (e.g. compilation options).

15.9.1.4 Action ALC_TAT.1.1E

15.9.1.4.1 General

CC Part 3 ALC_TAT.1.1C: *Each development tool used for implementation shall be well-defined.*

15.9.1.4.2 Work unit ALC_TAT.1-1

The evaluator ***shall examine*** the development tool documentation provided to determine that each development tools is well-defined.

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.

CC Part 3 ALC_TAT.1.2C: *The documentation of each development tool shall unambiguously define the meaning of all statements as well as all conventions and directives used in the implementation.*

15.9.1.4.3 Work unit ALC_TAT.1-2

The evaluator ***shall examine*** the documentation of each development tool to determine that it unambiguously defines the meaning of all statements as well as all conventions and directives used in the implementation.

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 should 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 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.

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.

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:

- 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.
- Aliasing (allowing the same piece of memory to be referenced in different ways) is a common source of ambiguity problems.
- Exception handling (e.g. what happens after memory exhaustion or stack overflow) is often poorly defined.

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.

The implementation standards description should include at least following:

- How to implement TOE so as to avoid problematic constructs inherent in the programming language used and/or in the development tool
- How to implement TOE so as to avoid unwanted behaviour from security perspective (or vulnerabilities) introduced by the development tool. Note that sometimes developers face the situation that the code generated by the development tool used by the developer does not fulfil the desired security property.
- How to implement TOE so as to meet the rules imposed by third party developers.

Note that information required under ADV_COMP.1.1C will be a part of the rules when composite evaluation approach is selected.

It is not sufficient to describe just naming rules which do no harm to the security in the light of modern compilers.

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.

The development tool documentation should define all conventions and directives used in the implementation.

CC Part 3 ALC_TAT.1.3C: *The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.*

15.9.1.4.4 Work unit 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.

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.

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).

15.9.2 Evaluation of sub-activity (ALC_TAT.2)

15.9.2.1 Objectives

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, and whether implementation standards have been applied.

15.9.2.2 Input

The evaluation evidence for this sub-activity is:

a) the development tool documentation;

b) the implementation standards description;

c) the provided implementation representation of the TSF.

15.9.2.3 Application notes

This work may be performed in parallel with the evaluation activities under ADV_IMP, specifically with regard to determining the use of features in the tools that will affect the object code (e.g. compilation options).

15.9.2.4 Action ALC_TAT.2.1E

15.9.2.4.1 General

CC Part 3 ALC_TAT.2.1C: *Each development tool used for implementation shall be well-defined.*

15.9.2.4.2 Work unit ALC_TAT.2-1

The evaluator ***shall examine*** the development tool documentation provided to determine that each development tool is well-defined.

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.

CC Part 3 ALC_TAT.2.2C: *The documentation of each development tool shall unambiguously define the meaning of all statements as well as all conventions and directives used in the implementation.*

15.9.2.4.3 Work unit ALC_TAT.2-2

The evaluator ***shall examine*** the documentation of each development tool to determine that it unambiguously defines the meaning of all statements as well as all conventions and directives used in the implementation.

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 should 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 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.

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.

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.

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.

The implementation standards description should include at least following:

a) How to implement TOE so as to avoid problematic constructs inherent in the programming language used and/or in the development tool;

b) How to implement TOE so as to avoid unwanted behaviour from security perspective (or vulnerabilities) introduced by the development tool. Note that sometimes developers face the situation that the code generated by the development tool used by the developer does not fulfil the desired security property;

c) How to implement TOE so as to meet the rules imposed by third party developers.

Note that information required under ADV_COMP.1.1C will be a part of the rules when composite evaluation approach is selected.

It is not sufficient to describe just naming rules which do no harm to the security in the light of modern compilers.

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.

The development tool documentation should define all conventions and directives used in the implementation.

CC Part 3 ALC_TAT.2.3C: *The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.*

15.9.2.4.4 Work unit ALC_TAT.2-3

The evaluator ***shall examine*** the development tool documentation to determine that it unambiguously defines the meaning of all implementation-dependent options.

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.

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).

15.9.2.5 Action ALC_TAT.2.2E

15.9.2.5.1 Work unit ALC_TAT.2-4

The evaluator ***shall examine*** aspects of the implementation process to determine that documented implementation standards have been applied.

This work unit requires the evaluator to analyse the provided implementation representation of the TOE to determine whether the documented implementation standards have been applied.

The evaluator should verify that constructs excluded by the documented standard are not used.

Additionally, the evaluator should verify the developer's procedures which ensure the application of the defined standards within the design and implementation process of the TOE. Therefore, 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 defined standards;

b) examine documentary evidence of application of procedures describing the use of defined standards;

c) interview development staff to check awareness of the application of defined standards and procedures.

A development site visit is a useful means of gaining confidence in the procedures being used. Any decision not to make such a visit should be determined in consultation with the evaluation authority.

The evaluator compares the provided implementation representation with the description of the applied implementation standards and verifies their use.

At this level it is not required that the complete provided implementation representation of the TSF is based on implementation standards, but only those parts that are developed by the TOE developer. The evaluator may consult the configuration list required by the CM scope (ALC_CMS) to get the information which parts are developed by the TOE developer, and which by third party developers.

If the referenced implementation standards are not applied for at least parts of the provided implementation representation, the evaluator action related to this work unit is assigned a fail verdict.

Note that parts of the TOE which are not TSF relevant do not need to be examined.

This work unit may be performed in conjunction with the evaluation activities under ADV_IMP.

15.9.3 Evaluation of sub-activity (ALC_TAT.3)

15.9.3.1 Objectives

The objective of this sub-activity is to determine whether the developer and their subcontractors have used well-defined development tools (e.g. programming languages or computer-aided design (CAD) systems) that yield consistent and predictable results, and whether implementation standards have been applied.

15.9.3.2 Input

The evaluation evidence for this sub-activity is:

a) the development tool documentation;

b) the implementation standards description;

c) the provided implementation representation of the TSF.

15.9.3.3 Application notes

This work may be performed in parallel with the evaluation activities under ADV_IMP, specifically with regard to determining the use of features in the tools that will affect the object code (e.g. compilation options).

15.9.3.4 Action ALC_TAT.3.1E

15.9.3.4.1 General

CC Part 3 ALC_TAT.3.1C: *Each development tool used for implementation shall be well-defined.*

15.9.3.4.2 Work unit ALC_TAT.3-1

The evaluator ***shall examine*** the development tool documentation provided to determine that each development tool is well-defined.

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.

At this level, the documentation of development tools used by third party contributors to the TOE has to be included in the evaluator's examination.

CC Part 3 ALC_TAT.3.2C: *The documentation of each development tool shall unambiguously define the meaning of all statements as well as all conventions and directives used in the implementation.*

15.9.3.4.3 Work unit ALC_TAT.3-2

The evaluator ***shall examine*** the documentation of each development tool to determine that it unambiguously defines the meaning of all statements as well as all conventions and directives used in the implementation.

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 should 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 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.

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.

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.

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.

The implementation standards description should include at least following:

a) How to implement TOE so as to avoid problematic constructs inherent in the programming language used and/or in the development tool

b) How to implement TOE so as to avoid unwanted behaviour from security perspective (or vulnerabilities) introduced by the development tool. Note that sometimes developers face the situation that the code generated by the development tool used by the developer does not fulfil the desired security property.

c) How to implement TOE so as to meet the rules imposed by third party developers

Note that information required under ADV_COMP.1.1C will be a part of the rules when composite evaluation approach is selected.

It is not sufficient to describe just naming rules which do no harm to the security in the light of modern compilers.

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.

The development tool documentation should define all conventions and directives used in the implementation.

At this level, the documentation of development tools used by third party contributors to the TOE has to be included in the evaluator's examination.

CC Part 3 ALC_TAT.3.3C: *The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.*

15.9.3.4.4 Work unit ALC_TAT.3-3

The evaluator ***shall examine*** the development tool documentation to determine that it unambiguously defines the meaning of all implementation-dependent options.

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.

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).

At this level, the documentation of development tools used by third party contributors to the TOE has to be included in the evaluator's examination.

15.9.3.5 Action ALC_TAT.3.2E

15.9.3.5.1 Work unit ALC_TAT.3-4

The evaluator ***shall examine*** aspects of the implementation process to determine that documented implementation standards have been applied.

This work unit requires the evaluator to analyse the provided implementation representation of the TOE to determine whether the documented implementation standards have been applied.

The evaluator should verify that constructs excluded by the documented standard are not used.

Additionally, the evaluator should verify the developer's procedures which ensure the application of the defined standards within the design and implementation process of the TOE. Therefore, 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 defined standards;

b) examine documentary evidence of application of procedures describing the use of defined standards;

c) interview development staff to check awareness of the application of defined standards and procedures.

A development site visit is a useful means of gaining confidence in the procedures being used. Any decision not to make such a visit should be determined in consultation with the evaluation authority.

The evaluator compares the provided implementation representation with the description of the applied implementation standards and verifies their use.

At this level it is required that the complete provided implementation representation of the TSF is based on implementation standards, including third party contributions. This may require the evaluator to visit the sites of contributors. The evaluator may consult the configuration list required by the CM scope (ALC_CMS) to see who has developed which part of the TOE.

Note that parts of the TOE which are not TSF relevant do not need to be examined.

This work unit may be performed in conjunction with the evaluation activities under ADV_IMP.

15.10 Integration of composition parts and consistency check of delivery procedures (ALC_COMP)

15.10.1 General

The composite-specific work units defined here are intended to be integrated as refinements to the evaluation activities of the ALC class listed in the following table. The other activities of ALC class do not require composite-specific work units.

Table 3 — ALC_COMP

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
ALC_CMS	ALC_CMS.1.1E	ALC_CMS.1-2	ALC_COMP.1-1
AGD_PRE	AGD_PRE.1.1E	AGD_PRE.1-2	ALC_COMP.1-2
ALC_CMC	ALC_CMC.4.1E	ALC_CMC.4-10	ALC_COMP.1-2

NOTE If the level of the assurance requirement chosen is higher than those identified in this table, the composite-specific work unit is also applicable.

15.10.2 Evaluation of sub-activity (ALC_COMP.1)

15.10.2.1 Objectives

The aim of this activity is to determine whether:

- the correct version of the dependent component is installed onto/ embedded into the correct version of the related base component, and

- the preparative guidance procedures of the base component developer and the dependent component developer are compatible with the acceptance procedures of the composite product integrator.

15.10.2.2 Application notes

The composite product evaluator shall verify that the correct version of the dependent component under evaluation has been installed onto / embedded into the evaluated version of the related base component of the composite product.

The composite product evaluation sponsor shall ensure that appropriate evidence generated by the composite product integrator is available for the composite product evaluator. This evidence may include, amongst other items, the configuration list of the base component developer (e.g. provided within his acknowledgement statement).

The composite product evaluator shall verify that the delivery procedures of the base component developer and the dependent component developer are compatible with the acceptance procedures used by the composite product integrator.

The composite product evaluator shall verify that all configuration parameters prescribed by the base component developer and the dependent component developer (e.g. pre-personalization data, pre-personalisation scripts) are used by the composite product integrator.

The composite product evaluation sponsor shall ensure that appropriate evidence generated by the composite product integrator is available for the composite product evaluator. This evidence may include, amongst other items, the element of evidence for the dependent component reception, acceptance and parameterisation by the base component developer (e.g. in form of his acknowledgement statement).

15.10.2.3 Action ALC_COMP.1.1E

15.10.2.3.1 General

CC Part 3 ALC_COMP.1.1C: *The components configuration evidence shall show that the evaluated version of the dependent component has been installed onto / embedded into the evaluated version of the related base component.*

15.10.2.3.2 Work unit ALC_COMP.1-1

The evaluator ***shall examine*** the evidence that the evaluated version of the dependent component has been installed onto / embedded into the correct evaluated version of the related base component.

NOTE From the perspective of the composite evaluation it is important that the correct evaluated version of the dependent component and base component are used.

The AGD_PRE documentation of the base component provided by the base component developer contains requirements for the secure acceptance procedure of the base component and security measures to which the dependent component developer or composite product integrator shall adhere. The dependent component developer shall provide evidence that (if applicable), these requirements are followed up and the required security measures are implemented.

The special composite product evaluator activity is to check the evidence of the version correctness for both parts of the composite product and that the secure acceptance and installation of the base component has been performed.

For the base component, the evaluator shall determine that the actual identification of the base component is commensurate with the expected evaluated data of the base component evaluation as part of following up on the procedures as specified in the AGD_PRE of the base component.

For the dependent component, the relevant task is trivial due to the fact that the composite product evaluator shall perform this task in the context of the assurance family ALC_CMS.

Components identification evidence can be supplied in two different ways: technical and organisational.

A technical evidence of version correctness is being generated by the composite product itself: the base component and the dependent component return, in each case, strings containing unambiguous version numbers as answers to the respective commands. For example, it can be the return string of a command or the hard copy of the Windows Information (like 'About'); in case of smart cards it can be an appropriate ATR.

Technical evidence of version correctness for hardware can also be supplied, if applicable, by reading out the unambiguous inscription on its surface. Note that there is no physical indication existing on most smart card's microcontrollers.

Technical evidence is recommended to be provided.

An organisational evidence of version correctness is being generated by the composite product integrator on the basis of his configuration lists containing unambiguous version information of the base component and the dependent component having been composed into the final composite product.

For example, in case of smart cards it can be an acknowledgement statement (e.g. configuration list) of the underlying platform (base component) manufacturer to the application software (dependent component) manufacturer containing the evidence for the versions of the platform, any embedded software and its pre-personalisation parameters.

Organisational evidence is always possible and, hence, shall be provided.

The result of this work unit shall be integrated to the result of ALC_CMS.1.1E / ALC_CMS.1-2 (or the equivalent higher components if a higher assurance level is selected).

15.10.2.4 Action ALC_COMP.1.2E

15.10.2.4.1 General

CC Part 3 ALC_COMP.1.2C: *The components configuration evidence shall show that:*

a) *The evidence for delivery and acceptance compatibility shall show that the delivery procedures of the base component developer and the dependent component developer are compatible with the acceptance procedures of the composite product integrator.*
b) *The evidence shall show that preparative guidance procedures prescribed by the base component developer and the dependent component developer are either actually being used by the composite product integrator or compatible with the composite product integrator guidance and do not contradict each other.*

15.10.2.4.2 Work unit ALC_COMP.1-2

The evaluator ***shall examine*** the acceptance procedures of the composite product integrator, the delivery procedures of the base component developer and the dependent component developer to see that they are compatible and where necessary either applied by the composite product integrator or prescribed in the preparative guidance.

The general information of the preparative guidance requirements that amongst others includes configuration parameters is represented and shall be examined in the context of the assurance family AGD_PRE [1.2C]. The special evaluator activity is to examine the developer's evidence and to decide whether the composite product integrator appropriately treats this special subset of the preparative guidance requirements.

The evaluator shall examine this provided evidence which includes the check whether the delivery procedures of the base component developer and the dependent component developer are compatible with the acceptance procedures of the composite product integrator.

In the cases where the composite product integrator leaves preparative guidance requirements prescribed by the base component developer and the dependent component developer to the user, the composite product evaluator verifies that such requirements are presented in the preparative guidance of the composite evaluation.

For example, for a Java Card as a composite product, the Card Issuer shall set all parameters as prescribed by the Java Card Platform (base component) developer and the Applet (dependent component) developer while installing the Applet onto the Java Card Platform. Also, the Card Issuer shall verify that the package is byte-code-verified and has a valid digital signature.

The result of this work unit shall be integrated to the result of AGD_PRE.1.1E / AGD_PRE.1-2 and ALC_CMC.4.1E / ALC_CMC.4-10.

16 Class ATE: Tests

16.1 General

The goal of this activity is to determine whether the TOE behaves as described in the ST and as specified in the evaluation evidence (described in the ADV class). This determination is achieved through some combination of the developer's own functional testing of the TSF [Functional tests (ATE_FUN)] and independent testing the TSF by the evaluator [Independent testing (ATE_IND)]. At the lowest level of assurance, there is no requirement for developer involvement, so the only testing is conducted by the evaluator, using the limited available information about the TOE. Additional assurance is gained as the developer becomes increasingly involved both in testing and in providing additional information about the TOE, and as the evaluator increases the independent testing activities.

16.2 Application notes

Testing of the TSF is conducted by the evaluator and, in most cases, by the developer. The evaluator's testing efforts consist not only of creating and running original tests, but also of assessing the adequacy of the developer's tests and re-running a subset of them.

The evaluator analyses the developer's tests to determine the extent to which they are sufficient to demonstrate that TSFI [see Functional specification (ADV_FSP)] perform as specified, and to understand the developer's approach to testing. Similarly, the evaluator analyses the developer's tests to determine the extent to which they are sufficient to demonstrate the internal behaviour and properties of the TSF.

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 takes a testing approach that is different from that of the developer, particularly if the developer's tests have shortcomings.

To determine the adequacy of developer's test documentation or to create new tests, the evaluator needs to understand the desired expected behaviour of the TSF, both internally and as seen at the TSFI, in the context of the SFRs it is to satisfy. The evaluator may choose to divide the TSF and TSFI into subsets according to functional areas of the ST (audit subsystem, audit-related TSFI, authentication module, authentication-related TSFI, etc.) if they were not already divided in the ST, and focus on one subset of the TSF and TSFI at a time, examining the ST requirement and the relevant parts of the development and guidance documentation to gain an understanding of the way the TOE is expected to behave. This reliance upon the development documentation underscores the need for the dependencies on ADV by Coverage (ATE_COV) and Depth (ATE_DPT).

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 units across sub-activities. These application notes are used to minimise duplication of text between sub-activities.

16.2.1 Understanding the expected behaviour of the TOE

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.

As mentioned earlier, the evaluator may choose to subset the TSF and TSFI according to SFRs (audit, authentication, etc.) in the ST and focus on one subset at a time. The evaluator examines each ST requirement and the relevant parts of the functional specification and guidance documentation to gain an understanding of the way the related TSFI is expected to behave. Similarly, the evaluator examines the relevant parts of the TOE design and security architecture documentation to gain an understanding of the way the related modules or subsystems of the TSF are expected to behave.

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 TSFI being stimulated and its responses observed. Externally-visible functionality can be tested directly; however, in cases where functionality is not visible external to the TOE (for example, testing the residual information protection functionality), other means will need to be employed.

16.2.2 Testing vs. alternate approaches to verify the expected behaviour of functionality

In cases where it is impractical or inadequate to test specific functionality (where it provides no externally-visible TSFI), 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 can 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 claimed assurance requirements do not include availability of lower-level descriptions of the TOE modules, e.g. Evaluation of sub-activity (ADV_TDS.3), or implementation [Implementation representation (ADV_IMP)]**.** If evidence of developer integration or module testing is used in verifying the expected behaviour of a security functionality, care should be given to confirm that the testing evidence reflects the current implementation of the TOE. If the subsystems 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.

It should be emphasised 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.

16.2.3 Verifying the adequacy of tests

Test pre-requisites 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 pre-requisites for another test. The evaluator must determine that the pre-requisites are complete and appropriate in that they will not bias the observed test results towards the expected test results.

The test steps and expected results specify the actions and parameters to be applied to the TSFI 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 descriptions of the TSFI in the functional specification. This means that each characteristic of the TSFI behaviour explicitly described in the functional specification should have tests and expected results to verify that behaviour.

The overall aim of this testing activity is to determine that each subsystem, module, and TSFI has been sufficiently tested against the behavioural claims in the functional specification, TOE design, and architecture description. At the higher assurance levels, testing also includes bounds testing and negative testing. The test procedures will provide insight as to how the TSFIs, modules, and subsystems have been exercised by the developer during testing. The evaluator uses this information when developing additional tests to independently test the TSF.

16.3 Coverage (ATE_COV)

16.3.1 Evaluation of sub-activity (ATE_COV.1)

16.3.1.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested the TSFIs, and that the developer's test coverage evidence shows correspondence between the tests identified in the test documentation and the TSFIs described in the functional specification.

16.3.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test coverage evidence.

16.3.1.3 Application notes

The coverage analysis provided by the developer is required to show the correspondence between the tests provided as evaluation evidence and the functional specification. However, the coverage analysis need not demonstrate that all TSFI have been tested, or that all externally-visible interfaces to the TOE have been tested. Such shortcomings are considered by the evaluator during the independent testing [Evaluation of sub-activity (ATE_IND.2)] sub-activity.

16.3.1.4 Action ATE_COV.1.1E

16.3.1.4.1 General

CC Part 3 ATE_COV.1.1C: *The evidence of the test coverage shall show the correspondence between the tests in the test documentation and the TSFIs in the functional specification.*

16.3.1.4.2 Work unit 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 TSFIs described in the functional specification is accurate.

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.

16.3.2 Evaluation of sub-activity (ATE_COV.2)

16.3.2.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested all of the TSFIs, and that the developer's test coverage evidence shows correspondence between the tests identified in the test documentation and the TSFIs described in the functional specification.

16.3.2.2 Input

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test coverage analysis.

16.3.2.3 Action ATE_COV.2.1E

16.3.2.3.1 General

CC Part 3 ATE_COV.2.1C: *The analysis of the test coverage shall demonstrate the correspondence between the tests in the test documentation and the TSFIs in the functional specification.*

16.3.2.3.2 Work unit ATE_COV.2-1

The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the tests in the test documentation and the interfaces in the functional specification is accurate.

A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the interfaces presented in the test coverage analysis has to be unambiguous.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to interfaces in the functional specification.

16.3.2.3.3 Work unit ATE_COV.2-2

The evaluator ***shall examine*** the test plan to determine that the testing approach for each interface demonstrates the expected behaviour of that interface.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

16.3.2.3.4 Work unit 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 interface.

Guidance on this work units, as it pertains to the functional specification, can be found in:

- 15.2.3**,** Verifying the adequacy of tests.

CC Part 3 ATE_COV.2.2C: *The analysis of the test coverage shall demonstrate that all TSFIs in the functional specification have been tested.*

16.3.2.3.5 Work unit ATE_COV.2-4

The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the interfaces in the functional specification and the tests in the test documentation is complete.

All TSFIs 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. Incomplete coverage would be evident if an interface was identified in the functional specification and no test was mapped to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to interfaces in the functional specification.

16.3.3 Evaluation of sub-activity (ATE_COV.3)

16.3.3.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested all of the TSFIs exhaustively, and that the developer's test coverage evidence shows correspondence between the tests identified in the test documentation and the TSFIs described in the functional specification.

A particular objective of this component is to confirm that all parameters of all of the TSFIs have been tested.

16.3.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation;

d) the test coverage analysis.

16.3.3.3 Action ATE_COV.3.1E

16.3.3.3.1 General

CC Part 3 ATE_COV.3.1C: *The analysis of the test coverage shall demonstrate the correspondence between the tests in the test documentation and the TSFIs in the functional specification.*

16.3.3.3.2 Work unit ATE_COV.3-1

The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the tests in the test documentation and the interfaces in the functional specification is accurate.

A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the interfaces presented in the test coverage analysis has to be unambiguous.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to interfaces in the functional specification.

16.3.3.3.3 Work unit ATE_COV.3-2

The evaluator ***shall examine*** the test plan to determine that the testing approach for each interface demonstrates the expected behaviour of that interface.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

16.3.3.3.4 Work unit ATE_COV.3-3

The evaluator ***shall examine*** the test procedures to determine that the test prerequisites, test steps and expected result(s) adequately test each interface.

Guidance on this work unit, as it pertains to the functional specification, can be found in:

- 15.2.3 Verifying the adequacy of tests.

CC Part 3 ATE_COV.3.2C: *The analysis of the test coverage shall demonstrate that all TSFIs in the functional specification have been completely tested.*

16.3.3.3.5 Work unit ATE_COV.3-4

The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the interfaces in the functional specification and the tests in the test documentation is complete.

All TSFIs 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. Exhaustive specification testing of interfaces is required for this mapping. Incomplete coverage would be evident if an interface was identified in the functional specification and no test was mapped to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to interfaces in the functional specification.

16.3.3.3.6 Work unit ATE_COV.3-5

The evaluator ***shall examine*** the test coverage analysis to determine that the correspondence between the interfaces in the functional specification and the tests in the test documentation shows that all TSFIs were tested completely.

This means that the evaluator examines whether all aspects of purpose, method of use, parameters, parameter descriptions, actions and error messages for all TSFIs present in the functional specification are covered by the tests. Note that the level of detail present in the functional specification depends on the component of ADV_FSP chosen in the ST of the TOE.

The evaluator may conclude that the higher-level descriptions in the functional specification, like purpose or method of use, are implicitly covered, if coverage of lower-level descriptions like parameters, parameter descriptions, actions and error messages are covered. Therefore, in general it will only be necessary to confirm coverage on these lower levels.

The evaluator is reminded that (for example) coverage of all parameters does not necessarily mean coverage of every possible value a parameter may allow. However, every value for which a distinct qualitative behaviour of the TOE is expected, needs to be covered.

As an example: If one of the parameters of a function call is a two-byte value, which specifies the length of further parameters, only some typical values need to be tested. However, the evaluator will make sure that some specific cases (like the value zero or the maximal value) will be covered.

If the evaluator sees that a potential attacker might be able to invoke a TSFI with inconsistent parameter values (e. g. if one parameter specifies the length of a second parameter and it is possible to make the second parameter actually longer than the chosen value for the first parameter suggests) and this case is not covered by the developer's testing, the evaluator may decide either to test this during their activities in AVA_VAN or to require the developer to provide coverage also for this case.

Similar considerations as for parameters hold for error messages specified in the functional specification: Each error message, which belongs to a qualitatively distinct error case, needs to be covered by testing. Note, that there may be exceptions, for example error messages for errors, which cannot be provoked during testing. For such error messages other ways of coverage need to be found as discussed in 15.2.2, "Testing vs. alternate approaches to verify the expected behaviour of functionality".

Note that also the developer is allowed to use such alternative approaches to testing (e. g. checking something in the source code) in the coverage table. The evaluator has to examine, in this case, if this use of an alternative approach is acceptable (usually only in cases where testing is practically impossible).

16.4 Depth (ATE_DPT)

16.4.1 Evaluation of sub-activity (ATE_DPT.1)

16.4.1.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested the TSF subsystems against the TOE design and the security architecture description.

16.4.1.2 Input

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the test documentation;

f) the depth of testing analysis.

16.4.1.3 Action ATE_DPT.1.1E

16.4.1.3.1 General

CC Part 3 ATE_DPT.1.1C: *The analysis of the depth of testing shall demonstrate the correspondence between the tests in the test documentation and the TSF subsystems in the TOE design.*

16.4.1.3.2 Work unit ATE_DPT.1-1

The evaluator ***shall examine*** the depth of testing analysis to determine that the descriptions of the behaviour of TSF subsystems and of their interactions is included within the test documentation.

This work unit verifies the content of the correspondence between the tests and the descriptions in the TOE design. In cases where the description of the TSF's architectural soundness [in Security Architecture (ADV_ARC)] cites specific mechanisms, this work unit also verifies the correspondence between the tests and the descriptions of the behaviour of such mechanisms.

A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the behaviour/interaction presented in the depth-of coverage analysis has to be unambiguous.

When Evaluation of sub-activity (ATE_DPT.1) is combined with a component of TOE design (ADV_TDS), which includes descriptions at the module level, e.g. evaluation of sub-activity (ADV_TDS.3), the level of detail needed to map the test cases to the behaviour of the subsystems may require information from the module description to be used. This is because Evaluation of sub-activity (ADV_TDS.3) allows the description of details to be shifted from the subsystem level to the module level, or even to omit the subsystems altogether.

In any case, the required level of detail in the provided reference to the tested behaviour can be defined as "the level of detail required for the description of subsystem behaviour as defined by Evaluation of sub-activity (ADV_TDS.2) (in particular work unit ADV_TDS.2-4)". It states that a detailed description of the behaviour typically discusses how the functionality is provided, in terms of what key data and data structures represent; what control relationships exist within a subsystem and how these elements work together to provide the SFR-enforcing behaviour.

The evaluator is reminded that not all tests in the test documentation must map to a subsystem behaviour or interaction description.

16.4.1.3.3 Work unit ATE_DPT.1-2

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the behaviour of that subsystem as described in the TOE design.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

When Evaluation of sub-activity (ATE_DPT.1) is combined with a component of TOE design (ADV_TDS), which includes descriptions at the module level, e.g. Evaluation of sub-activity (ADV_TDS.3)**,** the level of detail needed to map the test cases to the behaviour of the subsystems may require information from the module description to be used. This is because Evaluation of sub-activity (ADV_TDS.3) allows the description of details to be shifted from the subsystem level to the module level, or even to omit the subsystems altogether.

In any case, the required level of detail in the provided reference to the tested behaviour can be defined as "the level of detail required for the description of subsystem behaviour as defined by Evaluation of sub-activity (ADV_TDS.2) (in particular work unit ADV_TDS.2-4)". It states that a detailed description of the behaviour typically discusses how the functionality is provided, in terms of what key data and data structures represent; what control relationships exist within a subsystem and how these elements work together to provide the SFR-enforcing behaviour.

If TSF subsystem interfaces are described, the behaviour of those subsystems may be tested directly from those interfaces. Otherwise, the behaviour of those subsystems is tested from the TSFI interfaces. Or a combination of the two may be employed. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the behaviour that is described in the TOE design.

16.4.1.3.4 Work unit ATE_DPT.1-3

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the interactions among subsystems as described in the TOE design.

While the previous work unit addresses behaviour of subsystems, this work unit addresses the interactions among subsystems.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

If TSF subsystem interfaces are described, the interactions with other subsystems may be tested directly from those interfaces. Otherwise, the interactions among subsystems must be inferred from the TSFI interfaces. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the interactions among subsystems that are described in the TOE design.

CC Part 3 ATE_DPT.1.2C: *The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.*

16.4.1.3.5 Work unit ATE_DPT.1-4

The evaluator ***shall examine*** the test procedures to determine that all descriptions of TSF subsystem behaviour and interaction are tested.

This work unit verifies the completeness of work unit ATE_DPT.1-1. All descriptions of TSF subsystem behaviour and of interactions among TSF subsystems that are provided in the TOE design have to be tested. Incomplete depth of testing would be evident if a description of TSF subsystem behaviour or of interactions among TSF subsystems was identified in the TOE design and no tests can be attributed to it.

When Evaluation of sub-activity (ATE_DPT.1) is combined with a component of TOE design (ADV_TDS), which includes descriptions at the module level, e.g. Evaluation of sub-activity (ADV_TDS.3), the level of detail needed to map the test cases to the behaviour of the subsystems may require information from the module description to be used. This is because Evaluation of sub-activity (ADV_TDS.3) allows the description of details to be shifted from the subsystem level to the module level, or even to omit the subsystems altogether.

In any case, the required level of detail in the provided reference to the tested behaviour can be defined as "the level of detail required for the description of subsystem behaviour as defined by Evaluation of sub-activity (ADV_TDS.2) (in particular work unit ADV_TDS.2-4)". It states that a detailed description of the behaviour typically discusses how the functionality is provided, in terms of what key data and data structures represent; what control relationships exist within a subsystem and how these elements work together to provide the SFR-enforcing behaviour.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to the subsystem behaviour or interaction description in the TOE design.

16.4.2 Evaluation of sub-activity (ATE_DPT.2)

16.4.2.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested all the TSF subsystems and SFR-enforcing modules against the TOE design and the security architecture description.

16.4.2.2 Input

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the test documentation;

f) the depth of testing analysis.

16.4.2.3 Action ATE_DPT.2.1E

16.4.2.3.1 General

CC Part 3 ATE_DPT.2.1C: *The analysis of the depth of testing shall demonstrate the correspondence between the tests in the test documentation and the TSF subsystems and SFR-enforcing modules in the TOE design.*

16.4.2.3.2 Work unit ATE_DPT.2-1

The evaluator ***shall examine*** the depth of testing analysis to determine that descriptions of the behaviour of TSF subsystems and of their interactions are included within the test documentation.

This work unit verifies the content of the correspondence between the tests and the descriptions in the TOE design. In cases where the description of the TSF's architectural soundness [in Security Architecture (ADV_ARC)] cites specific mechanisms, this work unit also verifies the correspondence between the tests and the descriptions of the behaviour of such mechanisms.

A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the behaviour/interaction presented in the depth-of coverage analysis has to be unambiguous.

The evaluator is reminded that not all tests in the test documentation must map to a subsystem behaviour or interaction description.

16.4.2.3.3 Work unit ATE_DPT.2-2

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the behaviour of that subsystem as described in the TOE design.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

If TSF subsystem interfaces are described, the behaviour of those subsystems may be tested directly from those interfaces. Otherwise, the behaviour of those subsystems is tested from the TSFI interfaces. Or a combination of the two may be employed. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the behaviour that is described in the TOE design.

16.4.2.3.4 Work unit ATE_DPT.2-3

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the interactions among subsystems as described in the TOE design.

While the previous work unit addresses behaviour of subsystems, this work unit addresses the interactions among subsystems.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

If TSF subsystem interfaces are described, the interactions with other subsystems may be tested directly from those interfaces. Otherwise, the interactions among subsystems must be inferred from the TSFI interfaces. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the interactions among subsystems that are described in the TOE design.

16.4.2.3.5 Work unit ATE_DPT.2-4

The evaluator ***shall examine*** the depth of testing analysis to determine that the interfaces of SFR-enforcing modules are included within the test documentation.

This work unit verifies the content of the correspondence between the tests and the descriptions in the TOE design. In cases where the description of the TSF's architectural soundness [in Security Architecture (ADV_ARC)] cites specific mechanisms at the modular level, this work unit also verifies the correspondence between the tests and the descriptions of the behaviour of such mechanisms.

A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the SFR-enforcing modules presented in the depth-of coverage analysis has to be unambiguous.

The evaluator is reminded that not all tests in the test documentation must map to the interfaces of SFR-enforcing modules.

16.4.2.3.6 Work unit ATE_DPT.2-5

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for each SFR-enforcing module interface demonstrates the expected behaviour of that interface.

While work unit ATE_DPT.2-2 addresses expected behaviour of subsystems, this work unit addresses expected behaviour of the SFR-enforcing module interfaces that are covered by ATE_DPT.2-4.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;

- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

Testing of an interface may be performed directly at that interface, or at the external interfaces, or a combination of both. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the interfaces. Specifically, the evaluator determines whether testing at the internal interfaces 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.

CC Part 3 ATE_DPT.2.2C: *The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.*

16.4.2.3.7 Work unit ATE_DPT.2-6

The evaluator ***shall examine*** the test procedures to determine that all descriptions of TSF subsystem behaviour and interaction are tested.

This work unit verifies the completeness of work unit ATE_DPT.2-1. All descriptions of TSF subsystem behaviour and of interactions among TSF subsystems that are provided in the TOE design have to be tested. Incomplete depth of testing would be evident if a description of TSF subsystem behaviour or of interactions among TSF subsystems was identified in the TOE design and no tests can be attributed to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to the subsystem behaviour or interaction description in the TOE design.

CC Part 3 ATE_DPT.2.3C: *The analysis of the depth of testing shall demonstrate that the SFR-enforcing modules in the TOE design have been tested.*

16.4.2.3.8 Work unit ATE_DPT.2-7

The evaluator ***shall examine*** the test procedures to determine that all interfaces of SFR-enforcing modules are tested.

This work unit verifies the completeness of work unit ATE_DPT.2-4. All interfaces of SFR-enforcing modules that are provided in the TOE design have to be tested. Incomplete depth of testing would be evident if any interface of any SFR-enforcing modules was identified in the TOE design and no tests can be attributed to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to an interface of an SFR-enforcing module in the TOE design.

16.4.3 Evaluation of sub-activity (ATE_DPT.3)

16.4.3.1 Objectives

The objective of this sub-activity is to determine whether the developer has tested the all the TSF subsystems and modules against the TOE design and the security architecture description.

16.4.3.2 Input

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the test documentation;

f) the depth of testing analysis.

16.4.3.3 Action ATE_DPT.3.1E

16.4.3.3.1 General

CC Part 3 ATE_DPT.3.1C: *The analysis of the depth of testing shall demonstrate the correspondence between the tests in the test documentation and the TSF subsystems and modules in the TOE design.*

16.4.3.3.2 Work unit ATE_DPT.3-1

The evaluator ***shall examine*** the depth of testing analysis to determine that descriptions of the behaviour of TSF subsystems and of their interactions are included within the test documentation.

This work unit verifies the content of the correspondence between the tests and the descriptions in the TOE design. A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the behaviour/interaction presented in the depth-of coverage analysis has to be unambiguous.

The evaluator is reminded that not all tests in the test documentation must map to a subsystem behaviour or interaction description.

16.4.3.3.3 Work unit ATE_DPT.3-2

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the behaviour of that subsystem as described in the TOE design.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

If TSF subsystem interfaces are provided, the behaviour of those subsystems may be performed directly from those interfaces. Otherwise, the behaviour of those subsystems is tested from the TSFI interfaces. Or a combination of the two may be employed. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the behaviour that is described in the TOE design.

16.4.3.3.4 Work unit ATE_DPT.3-3

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for the behaviour description demonstrates the interactions among subsystems as described in the TOE design.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

While the previous work unit addresses behaviour of subsystems, this work unit addresses the interactions among subsystems.

If TSF subsystem interfaces are provided, the interactions with other subsystems may be performed directly from those interfaces. Otherwise, the interactions among subsystems must be

inferred from the TSFI interfaces. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the interactions among subsystems that are described in the TOE design.

16.4.3.3.5 Work unit ATE_DPT.3-4

The evaluator ***shall examine*** the depth of testing analysis to determine that the interfaces of TSF modules are included within the test documentation.

This work unit verifies the content of the correspondence between the tests and the descriptions in the TOE design. A simple cross-table may be sufficient to show test correspondence. The identification of the tests and the behaviour/interaction presented in the depth-of coverage analysis has to be unambiguous.

The evaluator is reminded that not all tests in the test documentation must map to a subsystem behaviour or interaction description.

16.4.3.3.6 Work unit ATE_DPT.3-5

The evaluator ***shall examine*** the test plan, test prerequisites, test steps and expected result(s) to determine that the testing approach for each TSF module interface demonstrates the expected behaviour of that interface.

Guidance on this work unit can be found in:

- 15.2.1, Understanding the expected behaviour of the TOE;
- 15.2.2, Testing vs. alternate approaches to verify the expected behaviour of functionality.

Testing of an interface may be performed directly at that interface, or at the external interfaces, or a combination of both. Whatever strategy is used the evaluator will consider its appropriateness for adequately testing the interfaces. Specifically, the evaluator determines whether testing at the internal interfaces 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.

CC Part 3 ATE_DPT.3.2C: *The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.*

16.4.3.3.7 Work unit ATE_DPT.3-6

The evaluator ***shall examine*** the test procedures to determine that all descriptions of TSF subsystem behaviour and interaction are tested.

This work unit verifies the completeness of work unit ATE_DPT.3-1. All descriptions of TSF subsystem behaviour and of interactions among TSF subsystems that are provided in the TOE design have to be tested. Incomplete depth of testing would be evident if a description of TSF subsystem behaviour or of interactions among TSF subsystems was identified in the TOE design and no tests can be attributed to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to the subsystem behaviour or interaction description in the TOE design.

CC Part 3 ATE_DPT.3.3C: *The analysis of the depth of testing shall demonstrate that all TSF modules in the TOE design have been tested.*

16.4.3.3.8 Work unit ATE_DPT.3-7

The evaluator ***shall examine*** the test procedures to determine that all interfaces of all TSF modules are tested.

This work unit verifies the completeness of work unit ATE_DPT.3-4. All interfaces of TSF modules that are provided in the TOE design have to be tested. Incomplete depth of testing would be evident if any interface of any TSF module was identified in the TOE design and no tests can be attributed to it.

The evaluator is reminded that this does not imply that all tests in the test documentation must map to an interface of a TSF module in the TOE design.

16.4.4 Evaluation of sub-activity (ATE_DPT.4)

CC Part 3 ATE_DPT.4.1C: *The analysis of the depth of testing shall demonstrate the correspondence between the tests in the test documentation and the TSF subsystems and modules in the TOE design.*

CC Part 3 ATE_DPT.4.2C: *The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.*

CC Part 3 ATE_DPT.4.3C: *The analysis of the depth of testing shall demonstrate that all modules in the TOE design have been tested.*

CC Part 3 ATE_DPT.4.4C: *The analysis of the depth of testing shall demonstrate that the TSF operates in accordance with its implementation representation.*

There is no general guidance; the scheme should be consulted for guidance on this sub-activity.

16.5 Functional tests (ATE_FUN)

16.5.1 Evaluation of sub-activity (ATE_FUN.1)

16.5.1.1 Objectives

The objective of this sub-activity is to determine whether the developer correctly performed and documented the tests in the test documentation.

16.5.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation.

16.5.1.3 Application notes

The extent to which the test documentation is required to cover the TSF is dependent upon the coverage assurance component.

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 TSFI for which the developer's test results indicate that it might not perform as specified should be tested independently by the evaluator to determine whether or not it does.

16.5.1.4 Action ATE_FUN.1.1E

16.5.1.4.1 General

CC Part 3 ATE_FUN.1.1C: *The test documentation shall consist of test plans, expected test results and actual test results.*

16.5.1.4.2 Work unit ATE_FUN.1-1

The evaluator ***shall check*** that the test documentation includes test plans, expected test results and actual test results.

The evaluator checks that test plans, expected tests results and actual test results are included in the test documentation.

CC Part 3 ATE_FUN.1.2C: *The test plans shall identify the tests to be performed and describe the scenarios for performing each test. These scenarios shall include any ordering dependencies on the results of other tests.*

16.5.1.4.3 Work unit ATE_FUN.1-2

The evaluator ***shall examine*** the test plan to determine that it describes the scenarios for performing each test.

The evaluator determines that the test plan provides information about the test configuration being used: both on the configuration of the TOE and on any test equipment being used. This information should be detailed enough to ensure that the test configuration is reproducible.

The evaluator also determines that the test plan provides information about how to execute the test: any necessary automated set-up procedures (and whether they require privilege to run), inputs to be applied, how these inputs are applied, how output is obtained, any automated clean-up procedures (and whether they require privilege to run), etc. This information should be detailed enough to ensure that the test is reproducible.

The evaluator may wish to employ a sampling strategy when performing this work unit.

16.5.1.4.4 Work unit ATE_FUN.1-3

The evaluator ***shall examine*** the test plan to determine that the TOE test configuration is consistent with the ST.

The TOE referred to in the developer's test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The evaluator verifies that all test configurations identified in the developer test documentation are consistent with the ST. For example, the ST can define configuration options that must be set, which can have an impact upon what constitutes the TOE by including or excluding additional portions. The evaluator verifies that all such variations of the TOE are considered.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

The evaluator may wish to employ a sampling strategy when performing this work unit.

If this work unit is applied to a component TOE that can be used/integrated in a composed TOE (see Class ACO: Composition), the following will apply. In the instances that the component TOE under evaluation depends on other components in the operational environment to support their operation, the developer may wish to consider using the other component(s) that will be used in the composed TOE to fulfil the requirements of the operational environment as one of the test configurations. This will reduce the amount an additional testing that will be required for the composed TOE evaluation.

16.5.1.4.5 Work unit ATE_FUN.1-4

The evaluator ***shall examine*** the test plans to determine that sufficient instructions are provided for any ordering dependencies.

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 perform actions in a test that will result in the generation of audit records, before performing a test to consider the searching and sorting of those audit records. 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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

CC Part 3 ATE_FUN.1.3C: *The expected test results shall show the anticipated outputs from a successful execution of the tests.*

16.5.1.4.6 Work unit ATE_FUN.1-5

The evaluator ***shall examine*** the test documentation to determine that all expected tests results are included.

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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

CC Part 3 ATE_FUN.1.4C: *The actual test results shall be consistent with the expected test results.*

16.5.1.4.7 Work unit ATE_FUN.1-6

The evaluator ***shall check*** that the actual test results in the test documentation are consistent with the expected test results in the test documentation.

A comparison of the actual and expected test results provided by the developer will reveal any inconsistencies between the results. 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 synthesise the actual data.

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.

It should be noted that the description of the process used to reduce or synthesise 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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

16.5.1.4.8 Work unit ATE_FUN.1-7

The evaluator ***shall report*** the developer testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause regarding the developer testing effort is:

a) TOE test configurations. The particular configurations of the TOE that were tested, including whether any privileged code was required to set up the test or clean up afterwards;

b) testing approach. An account of the overall developer testing strategy employed;

c) testing results. A description of the overall developer testing results.

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.

16.5.2 Evaluation of sub-activity (ATE_FUN.2)

16.5.2.1 Objectives

The objective of this sub-activity is to determine whether the developer correctly performed and documented the tests in the test documentation and to ensure that testing is structured such as to avoid circular arguments about the correctness of the interfaces being tested.

16.5.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the test documentation.

16.5.2.3 Application notes

Although the test procedures may state pre-requisite initial test conditions in terms of ordering of tests, they may not provide a rationale for the ordering. An analysis of test ordering, which provides this rationale, is an important factor in determining the adequacy of testing, as there is a possibility of faults being concealed by the ordering of tests.

16.5.2.4 Action ATE_FUN.2.1E

16.5.2.4.1 General

CC Part 3 ATE_FUN.2.1C: *The test documentation shall consist of test plans, expected test results and actual test results.*

16.5.2.4.2 Work unit ATE_FUN.2-1

The evaluator ***shall check*** that the test documentation includes test plans, expected test results and actual test results.

The evaluator checks that test plans, expected tests results and actual test results are included in the test documentation.

CC Part 3 ATE_FUN.2.2C: *The test plans shall identify the tests to be performed and describe the scenarios for performing each test. These scenarios shall include any ordering dependencies on the results of other tests.*

16.5.2.4.3 Work unit ATE_FUN.2-2

The evaluator ***shall examine*** the test plan to determine that it describes the scenarios for performing each test.

The evaluator determines that the test plan provides information about the test configuration being used: both on the configuration of the TOE and on any test equipment being used. This information should be detailed enough to ensure that the test configuration is reproducible.

The evaluator also determines that the test plan provides information about how to execute the test: any necessary automated set-up procedures (and whether they require privilege to run), inputs to be applied, how these inputs are applied, how output is obtained, any automated clean-up procedures (and whether they require privilege to run), etc. This information should be detailed enough to ensure that the test is reproducible.

The evaluator may wish to employ a sampling strategy when performing this work unit.

16.5.2.4.4 Work unit ATE_FUN.2-3

The evaluator ***shall examine*** the test plan to determine that the TOE test configuration is consistent with the ST.

The TOE referred to in the developer's test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The evaluator verifies that all test configurations identified in the developer test documentation are consistent with the ST. For example, the ST can define configuration options that must be set, which can have an impact upon what constitutes the TOE by including or excluding additional portions. The evaluator verifies that all such variations of the TOE are considered.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

The evaluator may wish to employ a sampling strategy when performing this work unit.

If this work unit is applied to a component TOE that can be used/integrated in a composed TOE (see Class ACO: Composition), the following will apply. In the instances that the component TOE under evaluation depends on other components in the operational environment to support their operation, the developer may wish to consider using the other component(s) that will be used in the composed TOE to fulfil the requirements of the operational environment as one of the test configurations. This will reduce the amount an additional testing that will be required for the composed TOE evaluation.

16.5.2.4.5 Work unit ATE_FUN.2-4

The evaluator ***shall examine*** the test plans to determine that sufficient instructions are provided for any ordering dependencies.

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 perform actions in a test that will result in the generation of audit records, before performing a test to consider the searching and sorting of those audit records. 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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

CC Part 3 ATE_FUN.2.3C: *The expected test results shall show the anticipated outputs from a successful execution of the tests.*

16.5.2.4.6 Work unit ATE_FUN.2-5

The evaluator ***shall examine*** the test documentation to determine that all expected tests results are included.

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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

CC Part 3 ATE_FUN.2.4C: *The actual test results shall be consistent with the expected test results.*

16.5.2.4.7 Work unit ATE_FUN.2-6

The evaluator ***shall check*** that the actual test results in the test documentation are consistent with the expected test results in the test documentation.

A comparison of the actual and expected test results provided by the developer will reveal any inconsistencies between the results. 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 synthesise the actual data.

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.

It should be noted that the description of the process used to reduce or synthesise 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.

The evaluator may wish to employ a sampling strategy when performing this work unit.

16.5.2.4.8 Work unit ATE_FUN.2-7

The evaluator ***shall report*** the developer testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

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, including whether any privileged code was required to set up the test or clean up afterwards;

b) testing approach: An account of the overall developer testing strategy employed;

c) testing results: A description of the overall developer testing results.

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.

CC Part 3 ATE_FUN.2.5C: *The test documentation shall include an analysis of the test procedure ordering dependencies.*

16.5.2.4.9 Work unit ATE_FUN.2-8

The evaluator ***shall examine*** the analysis of the test procedure ordering dependencies to determine that a sufficient justification for the chosen ordering of test cases is given.

Usually the evaluator will generate a table of all cases, where the test documentation requires a certain ordering of the tests and will then examine if sufficient justification is given in any case, why testing in this ordering is adequate and sufficient.

As an example we assume that the TSF provide a random number generator, which needs to be initialised (for example with an adequate seed) before random numbers of a specified quality can be generated. In this case the evaluator will consider the following question:

Does the test documentation only describe an ordering of tests, where the initialisation is done before calling the function to generate a random number?

In this case the justification needs to show, why the developer expects that in the intended environment of the TOE the random number function will not be called without initialisation of the random number generator.

If for example the user guidance documentation includes a clear instruction that the random number generator needs to be initialised adequately before being called, this may be considered adequate as a justification. Note that the question of whether it can be plausibly assumed that

users will follow such instruction is covered by the evaluation activities for the classes ASE and AGD and needs not to be re-examined here.

If, on the other hand, the TOE provides an authentication protocol, which implicitly uses random numbers provided by the random number generator, and an attacker can therefore "call" the random number generator implicitly by simply trying to authenticate, and if neither the TOE nor the environment prevent an attacker from doing this even before the random number generator is initialised, a test case needs to show, what happens in such situation.

If, for example, instead of returning a "bad" random number, the random number function would return an error, when called without proper initialisation, it would be much better to include a test showing this secure behaviour instead of trying to justify why the functions are only tested in the usual order.

NOTE Even without ATE_FUN.2 an evaluator would be expected to look for potential vulnerabilities like the one described above. However, ATE_FUN.2.5C adds assurance by requiring the developer to give a systematic justification, why their chosen order of test cases doesn't hide such potential failures of security functions.

16.6 Independent testing (ATE_IND)

16.6.1 Evaluation of sub-activity (ATE_IND.1)

16.6.1.1 Objectives

The goal of this activity is to determine, by independently testing a subset of the TSFI, whether the TOE behaves as specified in the functional specification and guidance documentation.

16.6.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the operational user guidance;

d) the preparative user guidance;

e) the TOE suitable for testing.

16.6.1.3 Action ATE_IND.1.1E

16.6.1.3.1 General

CC Part 3 ATE_IND.1.1C: *The TOE shall be suitable for testing.*

16.6.1.3.2 Work unit 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.

The TOE provided by the developer should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in

accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

16.6.1.3.3 Work unit ATE_IND.1-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

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 Evaluation of sub-activity (AGD_PRE.1) 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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

16.6.1.4 Action ATE_IND.1.2E

16.6.1.4.1 Work unit ATE_IND.1-3

The evaluator ***shall devise*** a test subset.

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 interfaces as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few interfaces based on their perceived relevance and rigorously test these interfaces.

Typically, the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the interfaces using at least one test, but testing need not demonstrate exhaustive specification testing.

The evaluator, when selecting the subset of the interfaces to be tested, should consider the following factors:

- The number of interfaces from which to draw upon for the test subset. Where the TSF includes only a small number of relatively simple interfaces, it may be practical to rigorously test all of the interfaces. In other cases this may not be cost-effective, and sampling is required.
- 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.

The evaluator selects the interfaces 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:

- Significance of interfaces. Those interfaces more significant than others should be included in the test subset. One major factor of "significance" is the security-relevance [SFR-enforcing interfaces would be more significant than SFR-supporting interfaces, which are more

significant than SFR-non-interfering interfaces; see CC Part 3 Functional specification (ADV_FSP)]. The other major factor of "significance" is the number of SFRs mapping to this interface (as determined when identifying the correspondence between levels of abstraction in ADV).

- Complexity of the interface. Complex interfaces may require complex tests that impose onerous requirements on the developer or evaluator, which may not be conducive to cost-effective evaluations. Conversely, they 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.

- Implicit testing. Testing some interfaces may often implicitly test other interfaces, and their inclusion in the subset may maximise the number of interfaces 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.

- Types of interfaces (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

- Interfaces that give rise to features that are innovative or unusual. Where the TOE contains innovative or unusual features, which may feature strongly in marketing literature and guidance documents, the corresponding interfaces should be strong candidates for testing.

This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

16.6.1.4.2 Work unit 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.

With an understanding of the expected behaviour of the TSF, from the ST and the functional specification, the evaluator has to determine the most feasible way to test the interface. Specifically, the evaluator considers:

a) the approach that will be used, for instance, whether an external interface will be tested, or an internal interface using a test harness, or will an alternate test approach be employed (e.g. in exceptional circumstances, a code inspection, if the implementation representation is available);

b) the interface(s) that will be used to test 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 an interface (e.g. packet generators) or make observations of an interface (e.g. network analysers).

The evaluator may find it practical to test each interface using a series of test cases, where each test case will test a very specific aspect of expected behaviour.

The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant interface(s).

16.6.1.4.3 Work unit ATE_IND.1-5

The evaluator ***shall conduct*** testing.

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.

16.6.1.4.4 Work unit ATE_IND.1-6

The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the interface 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 interface;

e) instructions for observing the behaviour of the interface;

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.

The level of detail should be such that another evaluator can 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.

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.

16.6.1.4.5 Work unit ATE_IND.1-7

The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

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.

16.6.1.4.6 Work unit ATE_IND.1-8

The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities to gain some insight about the testing approach chosen,

amount of testing performed, TOE test configurations, and the overall results of the testing activity.

Information that would typically be found in the ETR subclause 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 interfaces that were tested during the evaluation and a justification for the size;

c) selection criteria for the interfaces that compose the subset. Brief statements about the factors considered when selecting interfaces for inclusion in the subset;

d) interfaces tested. A brief listing of the interfaces that merited inclusion in the subset;

e) 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.

16.6.2 Evaluation of sub-activity (ATE_IND.2)

16.6.2.1 Objectives

The goal 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 to gain confidence in the developer's test results by performing a sample of the developer's tests.

16.6.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design description;

d) the operational user guidance;

e) the preparative user guidance;

f) the configuration management documentation;

g) the test documentation;

h) the TOE suitable for testing.

16.6.2.3 Action ATE_IND.2.1E

16.6.2.3.1 General

CC Part 3 ATE_IND.2.1C: *The TOE shall be suitable for testing.*

16.6.2.3.2 Work unit 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.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

16.6.2.3.3 Work unit ATE_IND.2-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state.

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3**.**

CC Part 3 ATE_IND.2.2C: *The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.*

16.6.2.3.4 Work unit 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.

The set of resource used by the developer is documented in the developer test plan, as considered in the Functional tests (ATE_FUN) family. The resource set may include access to the relevant developer test environment, 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.

16.6.2.4 Action ATE_IND.2.2E

16.6.2.4.1 Work unit ATE_IND.2-4

The evaluator ***shall conduct*** testing using a sample of tests found in the developer test plan and procedures.

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 (see A.2).

All the developer tests can be traced back to specific interfaces. 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.2-6**.** Additionally, the evaluator may wish to employ a random sampling method to select developer tests to include in the sample.

16.6.2.4.2 Work unit ATE_IND.2-5

The evaluator ***shall check*** that all the actual test results are consistent with the expected test results.

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 can be resolved by a valid explanation and resolution of the inconsistencies by the developer.

If a satisfactory explanation or resolution cannot be reached, the evaluator's confidence in the developer's test results may be lessened and it may be necessary for the evaluator to increase the sample size to the extent that the 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 TOE by the developer (e.g., if the inconsistency is caused by incorrect behaviour) or to the developer's tests (e.g., if the inconsistency is caused by an incorrect test), or in the production of new tests by the evaluator.

16.6.2.5 Action ATE_IND.2.3E

16.6.2.5.1 Work unit ATE_IND.2-6

The evaluator ***shall devise*** a test subset.

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 interfaces as possible tested with little rigour. Another testing strategy would be to have the test subset contain a few interfaces based on their perceived relevance and rigorously test these interfaces.

Typically, the testing approach taken by the evaluator should fall somewhere between these two extremes. The evaluator should exercise most of the interfaces using at least one test, but testing need not demonstrate exhaustive specification testing.

The evaluator, when selecting the subset of the interfaces to be tested, should consider the following factors:

The developer test evidence. The developer test evidence consists of: the test documentation, the available test coverage analysis, and the available depth of testing analysis. The developer test evidence will provide insight as to how the TSF has 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:

a) Augmentation of developer testing for interfaces. The evaluator may wish to perform more of the same type of tests by varying parameters to more rigorously test the interface.

b) Supplementation of developer testing strategy for interfaces. The evaluator may wish to vary the testing approach of a specific interface by testing it using another test strategy.

c) The number of interfaces from which to draw upon for the test subset. Where the TSF includes only a small number of relatively simple interfaces, it may be practical to rigorously test all of them. In other cases, this may not be cost-effective, and sampling is required.

d) 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.

e) The evaluator selects the interfaces 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:

f) Rigour of developer testing of the interfaces. Those interfaces that the evaluator determines require additional testing should be included in the test subset.

g) Developer test results. If the results of developer tests cause the evaluator to doubt that an interface is not properly implemented, then the evaluator should include such interfaces in the test subset.

h) Significance of interfaces. Those interfaces more significant than others should be included in the test subset. One major factor of "significance" is the security-relevance (SFR-enforcing interfaces would be more significant than SFR-supporting interfaces, which are more significant than SFR-non-interfering interfaces; see CC Part 3 ADV_FSP). The other major factor of "significance" is the number of SFRs mapping to this interface (as determined when identifying the correspondence between levels of abstraction in ADV).

i) Complexity of interfaces. Interfaces that require complex implementation may require complex tests that impose onerous requirements on the developer or evaluator, which may not be conducive to cost-effective evaluations. Conversely, they 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.

j) Implicit testing. Testing some interfaces may often implicitly test other interfaces, and their inclusion in the subset may maximise the number of interfaces 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.

k) Types of interfaces (e.g. programmatic, command-line, protocol). The evaluator should consider including tests for all different types of interfaces that the TOE supports.

l) Interfaces that give rise to features that are innovative or unusual. Where the TOE contains innovative or unusual features, which may feature strongly in marketing literature and guidance documents, the corresponding interfaces should be strong candidates for testing.

This guidance articulates factors to consider during the selection process of an appropriate test subset, but these are by no means exhaustive.

16.6.2.5.2 Work unit ATE_IND.2-7

The evaluator ***shall produce*** test documentation for the test subset that is sufficiently detailed to enable the tests to be reproducible.

With an understanding of the expected behaviour of the TSF, from the ST, the functional specification, and the TOE design description, the evaluator has to determine the most feasible way to test the interface. Specifically, the evaluator considers:

a) the approach that will be used, for instance, whether an external interface will be tested, or 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 interface(s) that will be used to test 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 an interface (e.g. packet generators) or make observations of an interface (e.g. network analysers).

The evaluator may find it practical to test each interface using a series of test cases, where each test case will test a very specific aspect of expected behaviour of that interface.

The evaluator's test documentation should specify the derivation of each test, tracing it back to the relevant interface(s).

16.6.2.5.3 Work unit ATE_IND.2-8

The evaluator ***shall conduct*** testing.

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.

16.6.2.5.4 Work unit ATE_IND.2-9

The evaluator ***shall record*** the following information about the tests that compose the test subset:

a) identification of the interface 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 interface;

e) instructions for observing the interface;

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.

The level of detail should be such that another evaluator can 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.

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.

16.6.2.5.5 Work unit ATE_IND.2-10

The evaluator ***shall check*** that all actual test results are consistent with the expected test results.

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.

16.6.2.5.6 Work unit ATE_IND.2-11

The evaluator ***shall report*** in the ETR the evaluator testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause 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 interfaces that were tested during the evaluation and a justification for the size;

c) selection criteria for the interfaces that compose the subset. Brief statements about the factors considered when selecting interfaces for inclusion in the subset;

d) Interfaces tested. A brief listing of the interfaces 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.

16.6.3 Evaluation of sub-activity (ATE_IND.3)

CC Part 3 ATE_IND.3.1C: *The TOE shall be suitable for testing.*

CC Part 3 ATE_IND.3.2C: *The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.*

There is no general guidance; the scheme should be consulted for guidance on this sub-activity.

16.7 Composite functional testing (ATE_COMP)

16.7.1 General

The composite-specific work units defined here are intended to be integrated as refinements to the evaluation activities of the ATE class listed in the following table. The other activities of the ATE class do not require composite-specific work units.

Table 4 — ATE_COMP

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
ATE_COV	ATE_COV.1.1E	ATE_COV.1-1	ATE_COMP.1-1

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
ATE_FUN	ATE_FUN.1.1E	ATE_FUN.1-3	ATE_COMP.1-1

NOTE If the level of the assurance requirement chosen is higher than those identified in this table, the composite-specific work unit is also applicable.

16.7.2 Evaluation of sub-activity (ATE_COMP.1)

16.7.2.1 Objectives

The aim of this activity is to determine whether the composite product as a whole exhibits the properties necessary to satisfy the functional requirements of its composite product Security Target.

16.7.2.2 Application notes

A composite product can be tested by testing its components separately and by testing the integrated product. Separate testing means that its base component and the dependent component are being tested independently of each other. A lot of tests of the base component may have been performed within the scope of its accomplished evaluation. The dependent component may be tested on a simulator or an emulator, which represent a virtual machine.

Integration testing means that the composite product is being tested as it is: the dependent component is running together with the related base component.

Some dependent component functionality testing can only be performed on emulators, before its embedding/integration into the base component, as effectiveness of this testing may not be visible using the interfaces of the composite product. Nevertheless, functional testing of the composite product shall be performed also on composite product samples according to the description of the security functions of the composite product and using the standard approach as required by the relevant ATE assurance class. No additional developer's action is required here.

Since the amount, the coverage and the depth of the functional tests of the base component have already been validated by the base component evaluation, it is not necessary to re-perform these tasks in the composite evaluation. Please note that the *ETR for composite evaluation* does not provide any information on functional testing for the base component.

The behaviour of implementation of some SFRs can depend on properties of the base component as well as of the dependent component (e.g. correctness of the measures of the composite product to withstand a side channel attack or correctness of the implementation of tamper resistance against physical attacks). In such case the SFR implementation shall be tested on the final composite product, but not on a simulator or an emulator.

This activity focuses exclusively on testing of the composite product as a whole and represents merely partial efforts within the general test approach being covered by the assurance class ATE. These integration tests shall be specified and performed, whereby the approach of the standard assurance families of the class ATE shall be applied.

The correct behaviour of the base component-TSF being relevant for the composite product Security Target (corresponding to the group RP_SFR-SERV and RP_SFR-MECH in the work unit ADV_COMP.1-1), and absence of exploitable vulnerabilities (sufficient effectiveness) in the context of the base component Security Target, are confirmed by the validity of the base component evaluation (i.e. acceptance of the base component evaluation by the report of the base component evaluation).

The composite product evaluation sponsor shall ensure that the following is available for the composite product evaluator:

- composite product samples suitable for testing.

16.7.2.3 Action ATE_COMP.1.1E

16.7.2.3.1 General

CC Part 3 ATE_COMP.1.1C: *Content and presentation of the specification and documentation of the integration tests shall correspond to the standard* [10] *requirements of the assurance families ATE_FUN and ATE_COV.*

CC Part 3 ATE_COMP.1.2C: *The composite product provided shall be suitable for testing.*

16.7.2.3.2 Work unit ATE_COMP.1-1

The evaluator ***shall examine*** that the developer performed the integration tests for all SFRs having to be tested for the composite product as a whole.

In order to perform this work unit the evaluator shall analyse, for each SFR, whether it directly depends on security properties of the base component and of the dependent component. Then the evaluator shall verify that the integration tests performed by the developer cover at least all such SFRs.

If the assurance package chosen does not contain the families ATE_FUN and ATE_COV (e.g. EAL1), this work unit is not applicable.

The result of this work unit shall be integrated to the result of ATE_COV.1.1E / ATE_COV.1-1 and ATE_FUN.1.1E / ATE_FUN.1-3 (or the equivalent higher components if a higher assurance level is selected).

---

[10] i.e. as defined by the CEM (this document).

17 Class AVA: Vulnerability assessment

17.1 General

The purpose of the vulnerability assessment activity is to determine the exploitability of flaws or weaknesses in the TOE in the operational environment. This determination is based upon analysis of the evaluation evidence and a search of publicly available material by the evaluator and is supported by evaluator penetration testing.

17.2 Vulnerability analysis (AVA_VAN)

17.2.1 Evaluation of sub-activity (AVA_VAN.1)

17.2.1.1 Objectives

The objective of this sub-activity is to determine whether the TOE, in its operational environment, has easily identifiable exploitable vulnerabilities.

17.2.1.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the guidance documentation;

c) the TOE suitable for testing;

d) information publicly available to support the identification of potential vulnerabilities.

Other input for this sub-activity is:

a) current information regarding potential vulnerabilities (e.g. from an evaluation authority).

17.2.1.3 Application notes

The evaluator should consider performing additional tests as a result of potential vulnerabilities encountered during the conduct of other parts of the evaluation.

The use of the term guidance in this sub-activity refers to the operational guidance and the preparative guidance.

Potential vulnerabilities may be in information that is publicly available, 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 potential vulnerability is identifiable from information that is publicly available, it can be easily exploited.

17.2.1.4 Action AVA_VAN.1.1E

17.2.1.4.1 General

CC Part 3 AVA_VAN.1.1C: *The TOE shall be suitable for testing.*

17.2.1.4.2 Work unit AVA_VAN.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.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

17.2.1.4.3 Work unit AVA_VAN.1-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

17.2.1.5 Action AVA_VAN.1.2E

17.2.1.5.1 Work unit AVA_VAN.1-3

The evaluator ***shall examine*** sources of information publicly available to identify potential vulnerabilities in the TOE.

The evaluator examines the sources of information publicly available to support the identification of possible potential vulnerabilities in the TOE. There are many sources of publicly available information, which should be considered, e.g. mailing lists and security forums on the world wide web that report known vulnerabilities in specified technologies.

The evaluator should not constrain their consideration of publicly available information to the above but should consider any other relevant information available.

While examining the evidence provided the evaluator will use the information in the public domain to further search for potential vulnerabilities. Where the evaluators have identified areas of concern, the evaluator should consider information publicly available that relate to those areas of concern.

The availability of information that may be readily available to an attacker that helps to identify and facilitate attacks effectively operates to substantially enhance the attack potential of a given attacker. The accessibility of vulnerability information and sophisticated attack tools on the Internet makes it more likely that this information will be used in attempts to identify potential vulnerabilities in the TOE and exploit them. Modern search tools make such information easily

available to the evaluator, and the determination of resistance to published potential vulnerabilities and well-known generic attacks can be achieved in a cost-effective manner.

The search of the information publicly available should be focused on those sources that refer specifically to the product from which the TOE is derived. The extensiveness of this search should consider the following factors: TOE type, evaluator experience in this TOE type, expected attack potential and the level of ADV evidence available.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will report what actions were taken to identify potential vulnerabilities in the information publicly available. However, in this type of search, the evaluator may not be able to describe the steps in identifying potential vulnerabilities before the outset of the examination, as the approach may evolve as a result of findings during the search.

The evaluator will report the evidence examined in completing the search for potential vulnerabilities.

17.2.1.5.2 Work unit AVA_VAN.1-4

The evaluator ***shall record*** in the ETR the identified potential vulnerabilities that are candidates for testing and applicable to the TOE in its operational environment.

It may be identified that no further consideration of the potential vulnerability is required if for example the evaluator identifies that measures in the operational environment, either IT or non-IT, prevent exploitation of the potential vulnerability in that operational environment. For instance, restricting physical access to the TOE to authorized users only may effectively render a potential vulnerability to tampering unexploitable.

The evaluator records any reasons for exclusion of potential vulnerabilities from further consideration if the evaluator determines that the potential vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the TOE in its operational environment, which can be used as an input into penetration testing activities, shall be reported in the ETR by the evaluators.

17.2.1.6 Action AVA_VAN.1.3E

17.2.1.6.1 Work unit AVA_VAN.1-5

The evaluator ***shall devise*** penetration tests, based on the independent search for potential vulnerabilities.

The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, to the potential vulnerabilities identified during the search of the sources of information publicly available. Any current information provided to the evaluator by a third party (e.g. evaluation authority) regarding known potential vulnerabilities will be considered by the evaluator, together with any encountered potential vulnerabilities resulting from the performance of other evaluation activities.

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 potential vulnerability.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Basic attack potential. 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 potential vulnerability that is beyond Basic attack potential, this is reported in the ETR as a residual vulnerability.

17.2.1.6.2 Work unit AVA_VAN.1-6

The evaluator ***shall produce*** penetration test documentation for the tests based on the list of potential vulnerabilities in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the potential 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.

The evaluator prepares for penetration testing based on the list of potential vulnerabilities identified during the search of the public domain.

The evaluator is not expected to determine the exploitability for potential vulnerabilities beyond those for which a Basic attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a potential vulnerability that is exploitable only by an attacker with greater than Basic attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

With an understanding of the potential vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically, the evaluator considers:

- the TSFI or other TOE interface that will be used to stimulate the TSF and observe responses;
- 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);
- special test equipment that will be required to either stimulate a TSFI or make observations of a TSFI (although it is unlikely that specialist equipment would be required to exploit a potential vulnerability assuming a Basic attack potential);
- whether theoretical analysis should replace physical testing, particularly relevant where the results of an initial test can be extrapolated to demonstrate that repeated attempts of an attack are likely to succeed after a given number of attempts.

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 potential vulnerability.

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.

17.2.1.6.3 Work unit AVA_VAN.1-7

The evaluator ***shall conduct*** penetration testing.

The evaluator uses the penetration test documentation resulting from work unit AVA_VAN.1-5 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 learnt 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.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Basic attack potential. 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 potential vulnerability that is beyond Basic attack potential, this is reported in the ETR as a residual vulnerability.

17.2.1.6.4 Work unit AVA_VAN.1-8

The evaluator ***shall record*** the actual results of the penetration tests.

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 unexpected test results should be investigated. The impact on the evaluation should be stated and justified.

17.2.1.6.5 Work unit AVA_VAN.1-9

The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) TSFI penetration tested. A brief listing of the TSFI and other TOE interfaces that were the focus of the penetration testing;

c) verdict for the sub-activity. The overall judgement on the results of penetration testing.

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.

17.2.1.6.6 Work unit AVA_VAN.1-10

The evaluator ***shall examine*** the results of all penetration testing to determine that the TOE, in its operational environment, is resistant to an attacker possessing a Basic attack potential.

If the results reveal that the TOE, in its operational environment, has vulnerabilities exploitable by an attacker possessing less than Enhanced-Basic attack potential, then this evaluator action fails.

The guidance in B.2 should be used to determine the attack potential required to exploit a particular vulnerability and whether it can therefore be exploited in the intended environment. It may not be necessary for the attack potential to be calculated in every instance, only if there is some doubt as to whether or not the vulnerability can be exploited by an attacker possessing an attack potential less than Enhanced-Basic.

17.2.1.6.7 Work unit AVA_VAN.1-11

The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. evaluation methodology activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the SFR(s) not met;

c) a description;

d) whether it is exploitable in its operational environment or not (i.e. exploitable or residual).

e) the amount of time, level of expertise, level of knowledge of the TOE, level of opportunity and the equipment required to perform the identified vulnerabilities, and the corresponding values using Tables B.2 and B.3 of Annex B.

17.2.2 Evaluation of sub-activity (AVA_VAN.2)

17.2.2.1 Objectives

The objective of this sub-activity is to determine whether the TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing Basic attack potential.

17.2.2.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the guidance documentation;

f) the TOE suitable for testing;

g) information publicly available to support the identification of possible potential vulnerabilities.

The remaining implicit evaluation evidence for this sub-activity depends on the components that have been included in the assurance package. The evidence provided for each component is to be used as input in this sub-activity.

Other input for this sub-activity is:

a) current information regarding public domain potential vulnerabilities and attacks (e.g. from an evaluation authority).

17.2.2.3 Application notes

The evaluator should consider performing additional tests as a result of potential vulnerabilities encountered during other parts of the evaluation.

17.2.2.4 Action AVA_VAN.2.1E

17.2.2.4.1 General

CC Part 3 AVA_VAN.2.1C: *The TOE shall be suitable for testing.*

CC Part 3 AVA_VAN.2.2C: *The list of third party components shall include components provided by third parties, and that are part of the TOE or otherwise part of the TOE delivery.*

17.2.2.4.2 Work unit AVA_VAN.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.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

17.2.2.4.3 Work unit AVA_VAN.2-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

17.2.2.5 Action AVA_VAN.2.2E

17.2.2.5.1 Work unit AVA_VAN.2-3

The evaluator ***shall examine*** sources of information publicly available to identify potential vulnerabilities in the TOE.

The evaluator examines the sources of information publicly available to support the identification of possible potential vulnerabilities in the TOE. There are many sources of publicly available information which the evaluator should consider using items such as those available on the world wide web, including:

a) specialist publications (magazines, books);

b) research papers.

The evaluator should not constrain their consideration of publicly available information to the above, but should consider any other relevant information available.

While examining the evidence provided the evaluator will use the information in the public domain to further search for potential vulnerabilities. Where the evaluators have identified areas of concern, the evaluator should consider information publicly available that relate to those areas of concern.

The availability of information that may be readily available to an attacker that helps to identify and facilitate attacks may substantially enhance the attack potential of a given attacker. The accessibility of vulnerability information and sophisticated attack tools on the Internet makes it more likely that this information will be used in attempts to identify potential vulnerabilities in the TOE and exploit them. Modern search tools make such information easily available to the evaluator, and the determination of resistance to published potential vulnerabilities and well-known generic attacks can be achieved in a cost-effective manner.

The search of the information publicly available should be focused on those sources that refer specifically to the product from which the TOE is derived. The extensiveness of this search should consider the following factors: TOE type, evaluator experience in this TOE type, expected attack potential and the level of ADV evidence available.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will report what actions were taken to identify potential vulnerabilities in the evidence. However, in this type of search, the evaluator may not be able to describe the steps in identifying potential vulnerabilities before the outset of the examination, as the approach may evolve as a result of findings during the search.

The evaluator will report the evidence examined in completing the search for potential vulnerabilities. This selection of evidence may be derived from those areas of concern identified by the evaluator, linked to the evidence the attacker is assumed to be able to obtain, or according to another rationale provided by the evaluator.

17.2.2.6 Action AVA_VAN.2.3E

17.2.2.6.1 Work unit AVA_VAN.2-4

The evaluator ***shall conduct*** a search of ST, guidance documentation, functional specification, TOE design and security architecture description evidence to identify possible potential vulnerabilities in the TOE.

A search of the evidence should be completed whereby specifications and documentation for the TOE are analysed and then potential vulnerabilities in the TOE are hypothesised, or speculated. The list of hypothesised potential vulnerabilities is then prioritised on the basis of the estimated probability that a potential vulnerability exists and, assuming an exploitable 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.

The security architecture description provides the developer vulnerability analysis, as it documents how the TSF protects itself from interference from untrusted subjects and prevents the bypass of security enforcement functionality. Therefore, the evaluator should use this description of the protection of the TSF as a basis for the search for possible ways to undermine the TSF.

Subject to the SFRs the TOE is to meet in the operational environment, the evaluator's independent vulnerability analysis should consider generic potential vulnerabilities under each of the following headings:

generic potential vulnerabilities relevant for the type of TOE being evaluated, as may be supplied by the evaluation authority;

a) bypassing;

b) tampering;

c) direct attacks;

d) monitoring;

e) misuse.

Items b) to f) are explained in greater detail in Annex B.

The security architecture description should be considered in light of each of the above generic potential vulnerabilities. Each potential vulnerability should be considered to search for possible ways in which to defeat the TSF protection and undermine the TSF.

17.2.2.6.2 Work unit AVA_VAN.2-5

The evaluator ***shall record*** in the ETR the identified potential vulnerabilities that are candidates for testing and applicable to the TOE in its operational environment.

It may be identified that no further consideration of the potential vulnerability is required if for example the evaluator identifies that measures in the operational environment, either IT or non-IT, prevent exploitation of the potential vulnerability in that operational environment. For instance, restricting physical access to the TOE to authorized users only may effectively render a potential vulnerability to tampering unexploitable.

The evaluator records any reasons for exclusion of potential vulnerabilities from further consideration if the evaluator determines that the potential vulnerability is not applicable in the

operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the TOE in its operational environment, which can be used as an input into penetration testing activities, shall be reported in the ETR by the evaluators.

17.2.2.7 Action AVA_VAN.2.4E

17.2.2.7.1 Work unit AVA_VAN.2-6

The evaluator ***shall devise*** penetration tests, based on the independent search for potential vulnerabilities.

The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, to the potential vulnerabilities identified during the search of the sources of information publicly available. Any current information provided to the evaluator by a third party (e.g. evaluation authority) regarding known potential vulnerabilities will be considered by the evaluator, together with any encountered potential vulnerabilities resulting from the performance of other evaluation activities.

The evaluator is reminded that, as for considering the security architecture description in the search for vulnerabilities (as detailed in AVA_VAN.2-4), testing should be performed to confirm the architectural properties. This is likely to require negative tests attempting to disprove the properties of the security architecture. In developing the strategy for penetration testing, the evaluator will ensure that each of the major characteristics of the security architecture description are tested, either in functional testing (as considered in Clause 14) or evaluator penetration testing.

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 potential vulnerability.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Basic attack potential. 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 an exploitable vulnerability that is beyond Basic attack potential, this is reported in the ETR as a residual vulnerability.

Guidance on determining the necessary attack potential to exploit a potential vulnerability can be found in B.2.

Potential vulnerabilities hypothesised as exploitable only by attackers possessing Enhanced-Basic, 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.

Potential vulnerabilities hypothesised as exploitable by an attacker possessing a Basic 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.

17.2.2.7.2 Work unit AVA_VAN.2-7

The evaluator ***shall produce*** penetration test documentation for the tests based on the list of potential vulnerabilities in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

- identification of the potential vulnerability the TOE is being tested for;

- instructions to connect and setup all required test equipment as required to conduct the penetration test;
- instructions to establish all penetration test prerequisite initial conditions;
- instructions to stimulate the TSF;
- instructions for observing the behaviour of the TSF;
- descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;
- instructions to conclude the test and establish the necessary post-test state for the TOE.

The evaluator prepares for penetration testing based on the list of potential vulnerabilities identified during the search of the public domain and the analysis of the evaluation evidence.

The evaluator is not expected to determine the exploitability for potential vulnerabilities beyond those for which a Basic attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a potential vulnerability that is exploitable only by an attacker with greater than Basic attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

With an understanding of the potential vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically, the evaluator considers:

- the TSFI or other TOE interface that will be used to stimulate the TSF and observe responses (It is possible that the evaluator will need to use an interface to the TOE other than the TSFI to demonstrate properties of the TSF such as those described in the security architecture description (as required by ADV_ARC). It should the noted, that although these TOE interfaces provide a means of testing the TSF properties, they are not the subject of the test.);
- 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);
- special test equipment that will be required to either stimulate a TSFI or make observations of a TSFI (although it is unlikely that specialist equipment would be required to exploit a potential vulnerability assuming a Basic attack potential);
- whether theoretical analysis should replace physical testing, particularly relevant where the results of an initial test can be extrapolated to demonstrate that repeated attempts of an attack are likely to succeed after a given number of attempts.

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 potential vulnerability.

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.

17.2.2.7.3 Work unit AVA_VAN.2-8

The evaluator ***shall conduct*** penetration testing.

The evaluator uses the penetration test documentation resulting from work unit AVA_VAN.2-6 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 learnt 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.

Should penetration testing show that a hypothesised potential 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.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Basic attack potential. 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 an exploitable vulnerability that is beyond basic attack potential, this is reported in the ETR as a residual vulnerability.

17.2.2.7.4 Work unit AVA_VAN.2-9

The evaluator ***shall record*** the actual results of the penetration tests.

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 unexpected test results should be investigated. The impact on the evaluation should be stated and justified.

17.2.2.7.5 Work unit AVA_VAN.2-10

The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause regarding evaluator penetration testing efforts is:

- TOE test configurations. The particular configurations of the TOE that were penetration tested;
- TSFI penetration tested. A brief listing of the TSFI and other TOE interfaces that were the focus of the penetration testing;
- Verdict for the sub-activity. The overall judgement on the results of penetration testing.

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.

17.2.2.7.6 Work unit AVA_VAN.2-11

The evaluator ***shall examine*** the results of all penetration testing to determine that the TOE, in its operational environment, is resistant to an attacker possessing a Basic attack potential.

If the results reveal that the TOE, in its operational environment, has vulnerabilities exploitable by an attacker possessing less than an Enhanced-Basic attack potential, then this evaluator action fails.

The guidance in B.2 should be used to determine the attack potential required to exploit a particular vulnerability and whether it can therefore be exploited in the intended environment. It may not be necessary for the attack potential to be calculated in every instance, only if there is some doubt as to whether or not the vulnerability can be exploited by an attacker possessing an attack potential less than Enhanced-Basic.

17.2.2.7.7 Work unit AVA_VAN.2-12

The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. evaluation methodology activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the SFR(s) not met;

c) a description;

d) whether it is exploitable in its operational environment or not (i.e. exploitable or residual);

e) the amount of time, level of expertise, level of knowledge of the TOE, level of opportunity and the equipment required to perform the identified vulnerabilities, and the corresponding values using Tables B.2 and B.3 of B.2.

17.2.3 Evaluation of sub-activity (AVA_VAN.3)

17.2.3.1 Objectives

The objective of this sub-activity is to determine whether the TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing Enhanced-Basic attack potential.

17.2.3.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the implementation subset selected;

f) the guidance documentation;

g) the TOE suitable for testing;

h) information publicly available to support the identification of possible potential vulnerabilities;

i) the results of the testing of the basic design.

The remaining implicit evaluation evidence for this sub-activity depends on the components that have been included in the assurance package. The evidence provided for each component is to be used as input in this sub-activity.

Other input for this sub-activity is:

a) current information regarding public domain potential vulnerabilities and attacks (e.g. from an evaluation authority).

17.2.3.3 Application notes

During the conduct of evaluation activities the evaluator may also identify areas of concern. These are specific portions of the TOE evidence that the evaluator has some reservation about, although the evidence meets the requirements for the activity with which the evidence is associated. For example, a particular interface specification looks particularly complex, and therefore may be prone to error either in the development of the TOE or in the operation of the TOE. There is no potential vulnerability apparent at this stage, further investigation is required. This is beyond the bounds of encountered, as further investigation is required.

The focused approach to the identification of potential vulnerabilities is an analysis of the evidence with the aim of identifying any potential vulnerabilities evident through the contained information. It is an unstructured analysis, as the approach is not predetermined. Further guidance on focused vulnerability analysis can be found in B.1.4.2.2.

17.2.3.4 Action AVA_VAN.3.1E

17.2.3.4.1 General

CC Part 3 AVA_VAN.3.1C: *The TOE shall be suitable for testing.*

CC Part 3 AVA_VAN.3.2C: *The list of third party components shall include components provided by third parties, and that are part of the TOE or otherwise part of the TOE delivery.*

17.2.3.4.2 Work unit AVA_VAN.3-1

The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

17.2.3.4.3 Work unit AVA_VAN.3-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

17.2.3.5 Action AVA_VAN.3.2E

17.2.3.5.1 Work unit AVA_VAN.3-3

The evaluator ***shall examine*** sources of information publicly available to identify potential vulnerabilities in the TOE.

The evaluator examines the sources of information publicly available to support the identification of possible potential vulnerabilities in the TOE. There are many sources of publicly available information which the evaluator should consider using items such as those available on the world wide web, including:

a) specialist publications (magazines, books);

b) research papers;

c) conference proceedings.

The evaluator should not constrain their consideration of publicly available information to the above but should consider any other relevant information available.

While examining the evidence provided the evaluator will use the information in the public domain to further search for potential vulnerabilities. Where the evaluators have identified areas of concern, the evaluator should consider information publicly available that relate to those areas of concern.

The availability of information that may be readily available to an attacker that helps to identify and facilitate attacks may substantially enhance the attack potential of a given attacker. The accessibility of vulnerability information and sophisticated attack tools on the Internet makes it more likely that this information will be used in attempts to identify potential vulnerabilities in the TOE and exploit them. Modern search tools make such information easily available to the evaluator, and the determination of resistance to published potential vulnerabilities and well-known generic attacks can be achieved in a cost-effective manner.

The search of the information publicly available should be focused on those sources that refer to the technologies used in the development of the product from which the TOE is derived. The extensiveness of this search should consider the following factors: TOE type, evaluator experience in this TOE type, expected attack potential and the level of ADV evidence available.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will report what actions were taken to identify potential vulnerabilities in the evidence. However, in this type of search, the evaluator may not be able to describe the steps in identifying potential vulnerabilities before the outset of the examination, as the approach may evolve as a result of findings during the search.

The evaluator will report the evidence examined in completing the search for potential vulnerabilities. This selection of evidence may be derived from those areas of concern identified by the evaluator, linked to the evidence the attacker is assumed to be able to obtain, or according to another rationale provided by the evaluator.

17.2.3.6 Action AVA_VAN.3.3E

17.2.3.6.1 Work unit AVA_VAN.3-4

The evaluator ***shall conduct*** a focused search of ST, guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify possible potential vulnerabilities in the TOE.

A flaw hypothesis methodology needs to be used whereby specifications and development and guidance evidence are analysed and then potential vulnerabilities in the TOE are hypothesised, or speculated.

The evaluator uses the knowledge of the TOE design and operation gained from the TOE deliverables to conduct a flaw hypothesis to identify potential flaws in the development of the TOE and potential errors in the specified method of operation of the TOE.

The security architecture description provides the developer vulnerability analysis, as it documents how the TSF protects itself from interference from untrusted subjects and prevents the bypass of security enforcement functionality. Therefore, the evaluator should build upon the understanding of the TSF protection gained from the analysis of this evidence and then develop this in the knowledge gained from other development ADV evidence.

The approach taken is directed by areas of concern identified during examination of the evidence during the conduct of evaluation activities and ensuring a representative sample of the development and guidance evidence provided for the evaluation is searched.

For guidance on sampling see Annex A. This guidance should be considered when selecting the subset, giving reasons for:

a) the approach used in selection;

b) qualification that the evidence to be examined supports that approach.

The areas of concern may relate to the sufficiency of specific protection features detailed in the security architecture description.

The evidence to be considered during the vulnerability analysis may be linked to the evidence the attacker is assumed to be able to obtain. For example, the developer may protect the TOE design and implementation representations, so the only information assumed to be available to an attacker is the functional specification and guidance (publicly available). So, although the objectives for assurance in the TOE ensure the TOE design and implementation representation requirements are met, these design representations may only be searched to further investigate areas of concerns.

On the other hand, if the source is publicly available it would be reasonable to assume that the attacker has access to the source and can use this in attempts to attack the TOE. Therefore, the source should be considered in the focused examination approach.

The following indicates examples for the selection of the subset of evidence to be considered:

- For an evaluation where all levels of design abstraction from functional specification to implementation representation are provided, examination of information in the functional specification and the implementation representation may be selected, as the functional specification provides detail of interfaces available to an attacker, and the implementation representation incorporates the design decisions made at all other design abstractions. Therefore, the TOE design information will be considered as part of the implementation representation.

- Examination of a particular subset of information in each of the design representations provided for the evaluation.
- Coverage of particular SFRs through each of the design representations provided for the evaluation.
- Examination of each of the design representations provided for the evaluation, considering different SFRs within each design representations.
- Examination of aspects of the evidence provided for the evaluation relating to current potential vulnerability information the evaluator has received (e.g. from a scheme).

This approach to identification of potential vulnerabilities is to take an ordered and planned approach; applying a system to the examination. The evaluator is to describe the method to be used in terms of what evidence will be considered, the information within the evidence that is to be examined, the manner in which this information is to be considered and the hypothesis that is to be created.

The following provide some examples that a hypothesis may take:

a) consideration of malformed input for interfaces available to an attacker at the external interfaces;

b) examination of a key security mechanism cited in the security architecture description, such as process separation, hypothesising internal buffer overflows that may lead to degradation of separation;

c) search to identify any objects created in the TOE implementation representation that are then not fully controlled by the TSF, and can be used by an attacker to undermine SFRs.

For example, the evaluator may identify that interfaces are a potential area of weakness in the TOE and specify an approach to the search that "all interface specifications provided in the functional specification and TOE design will be searched to hypothesise potential vulnerabilities" and go on to explain the methods used in the hypothesis.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will report what actions were taken to identify potential vulnerabilities in the evidence. However, in this type of search, the evaluator may not be able to describe the steps in identifying potential vulnerabilities before the outset of the examination, as the approach may evolve as a result of findings during the search.

The evaluator will report the evidence examine in completing the search for potential vulnerabilities. This selection of evidence may be derived from those areas of concern identified by the evaluator, linked to the evidence the attacker is assumed to be able to obtain, or according to another rationale provided by the evaluator.

Subject to the SFRs the TOE is to meet in the operational environment, the evaluator's independent vulnerability analysis should consider generic potential vulnerabilities under each of the following headings:

a) generic potential vulnerabilities relevant for the type of TOE being evaluated, as may be supplied by the evaluation authority;

b) bypassing;

c) tampering;

d) direct attacks;

e) monitoring;

f) misuse.

Items b) to f) are explained in greater detail in Annex B.

The security architecture description should be considered in light of each of the above generic potential vulnerabilities. Each potential vulnerability should be considered to search for possible ways in which to defeat the TSF protection and undermine the TSF.

17.2.3.6.2 Work unit AVA_VAN.3-5

The evaluator ***shall record*** in the ETR the identified potential vulnerabilities that are candidates for testing and applicable to the TOE in its operational environment.

It may be identified that no further consideration of the potential vulnerability is required if for example the evaluator identifies that measures in the operational environment, either IT or non-IT, prevent exploitation of the potential vulnerability in that operational environment. For instance, restricting physical access to the TOE to authorized users only may effectively render a potential vulnerability to tampering unexploitable.

The evaluator records any reasons for exclusion of potential vulnerabilities from further consideration if the evaluator determines that the potential vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the TOE in its operational environment, which can be used as an input into penetration testing activities, shall be reported in the ETR by the evaluators.

17.2.3.7 Action AVA_VAN.3.4E

17.2.3.7.1 Work unit AVA_VAN.3-6

The evaluator ***shall devise*** penetration tests, based on the independent search for potential vulnerabilities.

The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, to the potential vulnerabilities identified during the search of the sources of information publicly available. Any current information provided to the evaluator by a third party (e.g. evaluation authority) regarding known potential vulnerabilities will be considered by the evaluator, together with any encountered potential vulnerabilities resulting from the performance of other evaluation activities.

The evaluator is reminded that, as for considering the security architecture description in the search for vulnerabilities (as detailed in AVA_VAN.3-4), testing should be performed to confirm the architectural properties. If requirements from ATE_DPT are included in the SARs, the developer testing evidence will include testing performed to confirm the correct implementation of any specific mechanisms detailed in the security architecture description. However, the developer testing will not necessarily include testing of all aspects of the architectural properties that protect the TSF, as much of this testing will be negative testing in nature, attempting to disprove the properties. In developing the strategy for penetration testing, the evaluator will ensure that all aspects of the security architecture description are tested, either in functional testing (as considered in Clause 14) or evaluator penetration testing.

It will probably be practical to carry out penetration test using a series of test cases, where each test case will test for a specific potential vulnerability.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required an Enhanced-Basic attack potential. 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 an exploitable vulnerability that is beyond Enhanced-Basic attack potential, this is reported in the ETR as a residual vulnerability.

Guidance on determining the necessary attack potential to exploit a potential vulnerability can be found in B.2.

Potential 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.

Potential vulnerabilities hypothesised as exploitable by an attacker possessing a Basic or Enhanced-Basic 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.

17.2.3.7.2 Work unit AVA_VAN.3-7

The evaluator ***shall produce*** penetration test documentation for the tests based on the list of potential vulnerabilities in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the potential 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.

The evaluator prepares for penetration testing based on the list of potential vulnerabilities identified during the search of the public domain and the analysis of the evaluation evidence.

The evaluator is not expected to determine the exploitability for potential vulnerabilities beyond those for which an Enhanced-Basic attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a potential vulnerability that is exploitable only by an attacker with greater than Enhanced-Basic attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

With an understanding of the potential vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically, the evaluator considers:

- the TSFI or other TOE interface that will be used to stimulate the TSF and observe responses (It is possible that the evaluator will need to use an interface to the TOE other than the TSFI

to demonstrate properties of the TSF such as those described in the security architecture description (as required by ADV_ARC). It should the noted, that although these TOE interfaces provide a means of testing the TSF properties, they are not the subject of the test.);

- 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);
- special test equipment that will be required to either stimulate a TSFI or make observations of a TSFI (although it is unlikely that specialist equipment would be required to exploit a potential vulnerability assuming an Enhanced-Basic attack potential);
- whether theoretical analysis should replace physical testing, particularly relevant where the results of an initial test can be extrapolated to demonstrate that repeated attempts of an attack are likely to succeed after a given number of attempts.

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 potential vulnerability.

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.

17.2.3.7.3 Work unit AVA_VAN.3-8

The evaluator ***shall conduct*** penetration testing.

The evaluator uses the penetration test documentation resulting from work unit AVA_VAN.3-6 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 learnt 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.

Should penetration testing show that a hypothesised potential 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.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required an Enhanced-Basic attack potential. 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 an exploitable vulnerability that is beyond Enhanced-Basic attack potential, this is reported in the ETR as a residual vulnerability.

17.2.3.7.4 Work unit AVA_VAN.3-9

The evaluator ***shall record*** the actual results of the penetration tests.

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 unexpected test results should be investigated. The impact on the evaluation should be stated and justified.

17.2.3.7.5 Work unit AVA_VAN.3-10

The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) TSFI penetration tested. A brief listing of the TSFI and other TOE interfaces that were the focus of the penetration testing;

c) Verdict for the sub-activity. The overall judgement on the results of penetration testing.

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.

17.2.3.7.6 Work unit AVA_VAN.3-11

The evaluator ***shall examine*** the results of all penetration testing to determine that the TOE, in its operational environment, is resistant to an attacker possessing an Enhanced-Basic attack potential.

If the results reveal that the TOE, in its operational environment, has vulnerabilities exploitable by an attacker possessing less than Moderate attack potential, then this evaluator action fails.

The guidance in B.2 should be used to determine the attack potential required to exploit a particular vulnerability and whether it can therefore be exploited in the intended environment. It may not be necessary for the attack potential to be calculated in every instance, only if there is some doubt as to whether or not the vulnerability can be exploited by an attacker possessing an attack potential less than Moderate.

17.2.3.7.7 Work unit AVA_VAN.3-12

The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. evaluation methodology activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the SFR(s) not met;

c) a description;

d) whether it is exploitable in its operational environment or not (i.e. exploitable or residual).

the amount of time, level of expertise, level of knowledge of the TOE, level of opportunity and the equipment required to perform the identified vulnerabilities, and the corresponding values using Tables B.2 and B.3 of B.2.

17.2.4 Evaluation of sub-activity (AVA_VAN.4)

17.2.4.1 Objectives

The objective of this sub-activity is to determine whether the TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing Moderate attack potential.

17.2.4.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the implementation representation;

f) the guidance documentation;

g) the TOE suitable for testing;

h) information publicly available to support the identification of possible potential vulnerabilities;

i) the results of the testing of the basic design.

The remaining implicit evaluation evidence for this sub-activity depends on the components that have been included in the assurance package. The evidence provided for each component is to be used as input in this sub-activity.

Other input for this sub-activity is:

a) current information regarding public domain potential vulnerabilities and attacks (e.g. from an evaluation authority).

17.2.4.3 Application notes

The methodical analysis approach takes the form of a structured examination of the evidence. This method requires the evaluator to specify the structure and form the analysis will take (i.e. the manner in which the analysis is performed is predetermined, unlike the focused analysis). The method is specified in terms of the information that will be considered and how/why it will be considered. Further guidance on methodical vulnerability analysis can be found in B.4.2.4.

17.2.4.4 Action AVA_VAN.4.1E

17.2.4.4.1 General

CC Part 3 AVA_VAN.4.1C: *The TOE shall be suitable for testing.*

CC Part 3 AVA_VAN.4.2C: *The list of third party components shall include components provided by third parties, and that are part of the TOE or otherwise part of the TOE delivery.*

17.2.4.4.2 Work unit AVA_VAN.4-1

The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

17.2.4.4.3 Work unit AVA_VAN.4-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

17.2.4.5 Action AVA_VAN.4.2E

17.2.4.5.1 Work unit AVA_VAN.4-3

The evaluator ***shall examine*** sources of information publicly available to identify potential vulnerabilities in the TOE.

The evaluator examines the sources of information publicly available to support the identification of possible potential vulnerabilities in the TOE. There are many sources of publicly available information which the evaluator should consider using items such as those available on the world wide web, including:

a) specialist publications (magazines, books);

b) research papers;

c) conference proceedings.

The evaluator should not constrain their consideration of publicly available information to the above, but should consider any other relevant information available.

While examining the evidence provided the evaluator will use the information in the public domain to further search for potential vulnerabilities. Where the evaluators have identified areas of concern, the evaluator should consider information publicly available that relate to those areas of concern.

The availability of information that may be readily available to an attacker that helps to identify and facilitate attacks may substantially enhance the attack potential of a given attacker. The accessibility of vulnerability information and sophisticated attack tools on the Internet makes it more likely that this information will be used in attempts to identify potential vulnerabilities in the TOE and exploit them. Modern search tools make such information easily available to the evaluator, and the determination of resistance to published potential vulnerabilities and well-known generic attacks can be achieved in a cost-effective manner.

The search of the information publicly available should be focused on those sources that refer to the technologies used in the development of the product from which the TOE is derived. The extensiveness of this search should consider the following factors: TOE type, evaluator experience in this TOE type, expected attack potential and the level of ADV evidence available.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will describe the approach to be taken to identify potential vulnerabilities in the publicly available material, detailing the search to be performed. This may be driven by factors such as areas of concern identified by the evaluator, linked to the evidence the attacker is assumed to be able to obtain. However, it is recognised that in this type of search the approach may further evolve as a result of findings during the search. Therefore, the evaluator will also report any actions taken in addition to those described in the approach to further investigate issues thought to lead to potential vulnerabilities, and will report the evidence examined in completing the search for potential vulnerabilities.

17.2.4.6 Action AVA_VAN.4.3E

17.2.4.6.1 Work unit AVA_VAN.4-4

The evaluator ***shall conduct*** a methodical analysis of ST, guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify possible potential vulnerabilities in the TOE.

Guidance on methodical vulnerability analysis is provided in B.1.4.2.3.

This approach to identification of potential vulnerabilities is to take an ordered and planned approach. A system is to be applied in the examination. The evaluator is to describe the method to be used in terms of the manner in which this information is to be considered and the hypothesis that is to be created.

A flaw hypothesis methodology needs to be used whereby the ST, development (functional specification, TOE design and implementation representation) and guidance evidence are analysed and then vulnerabilities in the TOE are hypothesised, or speculated.

The evaluator uses the knowledge of the TOE design and operation gained from the TOE deliverables to conduct a flaw hypothesis to identify potential flaws in the development of the TOE and potential errors in the specified method of operation of the TOE.

The security architecture description provides the developer vulnerability analysis, as it documents how the TSF protects itself from interference from untrusted subjects and prevents the bypass of security enforcement functionality. Therefore, the evaluator should build upon the understanding of the TSF protection gained from the analysis of this evidence and then develop this in the knowledge gained from other development ADV evidence.

The approach taken to the methodical search for vulnerabilities is to consider any areas of concern identified in the results of the evaluator's assessment of the development and guidance evidence. However, the evaluator should also consider each aspect of the security architecture

analysis to search for any ways in which the protection of the TSF can be undermined. It may be helpful to structure the methodical analysis on the basis of the material presented in the security architecture description, introducing concerns from other ADV evidence as appropriate. The analysis can then be further developed to ensure all other material from the ADV evidence is considered.

The following provide some examples of hypotheses that may be created when examining the evidence:

a) consideration of malformed input for interfaces available to an attacker at the external interfaces;

b) examination of a key security mechanism cited in the security architecture description, such as process separation, hypothesising internal buffer overflows that may lead to degradation of separation;

c) search to identify any objects created in the TOE implementation representation that are then not fully controlled by the TSF, and can be used by an attacker to undermine SFRs.

For example, the evaluator may identify that interfaces are a potential area of weakness in the TOE and specify an approach to the search that 'all interface specifications in the evidence provided will be searched to hypothesise potential vulnerabilities' and go on to explain the methods used in the hypothesis.

In addition, areas of concern the evaluator has identified during examination of the evidence during the conduct of evaluation activities. Areas of concern may also be identified during the conduct of other work units associated with this component, in particular AVA_VAN.4-7**,** AVA_VAN.4-5 and AVA_VAN.4-6 where the development and conduct of penetration tests may identify further areas of concerns for investigation, or potential vulnerabilities.

However, examination of only a subset of the development and guidance evidence or their contents is not permitted in this level of rigour. The approach description should provide a demonstration that the methodical approach used is complete, providing confidence that the approach used to search the deliverables has considered all of the information provided in those deliverables.

This approach to identification of potential vulnerabilities is to take an ordered and planned approach; applying a system to the examination. The evaluator is to describe the method to be used in terms of how the evidence will be considered; the manner in which this information is to be considered and the hypothesis that is to be created. This approach should be agreed with the evaluation authority, and the evaluation authority may provide detail of any additional approaches the evaluator should take to the vulnerability analysis and identify any additional information that should be considered by the evaluator.

Although a system to identifying potential vulnerabilities is predefined, the identification process may still be iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

Subject to the SFRs the TOE is to meet in the operational environment, the evaluator's independent vulnerability analysis should consider generic potential vulnerabilities under each of the following headings:

a) generic potential vulnerabilities relevant for the type of TOE being evaluated, as may be supplied by the evaluation authority;

b) bypassing;

c) tampering;

d) direct attacks;

e) monitoring;

f) misuse.

Items b) to f) are explained in greater detail in Annex B.

The security architecture description should be considered in light of each of the above generic potential vulnerabilities. Each potential vulnerability should be considered to search for possible ways in which to defeat the TSF protection and undermine the TSF.

17.2.4.6.2 Work unit AVA_VAN.4-5

The evaluator ***shall record*** in the ETR the identified potential vulnerabilities that are candidates for testing and applicable to the TOE in its operational environment.

It may be identified that no further consideration of the potential vulnerability is required if for example the evaluator identifies that measures in the operational environment, either IT or non-IT, prevent exploitation of the potential vulnerability in that operational environment. For instance, restricting physical access to the TOE to authorized users only may effectively render a potential vulnerability to tampering unexploitable.

The evaluator records any reasons for exclusion of potential vulnerabilities from further consideration if the evaluator determines that the potential vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the TOE in its operational environment, which can be used as an input into penetration testing activities, shall be reported in the ETR by the evaluators.

17.2.4.7 Action AVA_VAN.4.4E

17.2.4.7.1 Work unit AVA_VAN.4-6

The evaluator ***shall devise*** penetration tests, based on the independent search for potential vulnerabilities.

The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, to the potential vulnerabilities identified during the search of the sources of information publicly available. Any current information provided to the evaluator by a third party (e.g. evaluation authority) regarding known potential vulnerabilities will be considered by the evaluator, together with any encountered potential vulnerabilities resulting from the performance of other evaluation activities.

The evaluator is reminded that, as for considering the security architecture description in the search for vulnerabilities (as detailed in AVA_VAN.4-3), testing should be performed to confirm the architectural properties. If requirements from ATE_DPT are included in the SARs, the developer testing evidence will include testing performed to confirm the correct implementation of any specific mechanisms detailed in the security architecture description. However, the developer testing will not necessarily include testing of all aspects of the architectural properties that protect the TSF, as much of this testing will be negative testing in nature, attempting to disprove the properties. In developing the strategy for penetration testing, the evaluator will

ensure that all aspects of the security architecture description are tested, either in functional testing or evaluator penetration testing.

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 potential vulnerability.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Moderate attack potential. 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 an exploitable vulnerability that is beyond Moderate attack potential, this is reported in the ETR as a residual vulnerability.

Guidance on determining the necessary attack potential to exploit a potential vulnerability can be found in B.2.

Potential vulnerabilities hypothesised as exploitable by an attacker possessing a Moderate (or less) 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.

17.2.4.7.2 Work unit AVA_VAN.4-7

The evaluator ***shall produce*** penetration test documentation for the tests based on the list of potential vulnerabilities in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

- identification of the potential vulnerability the TOE is being tested for;
- instructions to connect and setup all required test equipment as required to conduct the penetration test;
- instructions to establish all penetration test prerequisite initial conditions;
- instructions to stimulate the TSF;
- instructions for observing the behaviour of the TSF;
- descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;
- instructions to conclude the test and establish the necessary post-test state for the TOE.

The evaluator prepares for penetration testing based on the list of potential vulnerabilities identified during the search of the public domain and the analysis of the evaluation evidence.

The evaluator is not expected to determine the exploitability for potential vulnerabilities beyond those for which a Moderate attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a potential vulnerability that is exploitable only by an attacker with greater than Moderate attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

With an understanding of the potential vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically the evaluator considers:

a) the TSFI or other TOE interface that will be used to stimulate the TSF and observe responses (It is possible that the evaluator will need to use an interface to the TOE other than the TSFI to demonstrate properties of the TSF such as those described in the security architecture

description (as required by ADV_ARC). It should the noted, that although these TOE interfaces provide a means of testing the TSF properties, they are not the subject of the test.);

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 TSFI or make observations of a TSFI;

d) whether theoretical analysis should replace physical testing, particularly relevant where the results of an initial test can be extrapolated to demonstrate that repeated attempts of an attack are likely to succeed after a given number of attempts.

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 potential vulnerability.

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.

17.2.4.7.3 Work unit AVA_VAN.4-8

The evaluator ***shall conduct*** penetration testing.

The evaluator uses the penetration test documentation resulting from work unit AVA_VAN.4-6 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 learnt 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.

Should penetration testing show that a hypothesised potential 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.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a Moderate attack potential. 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 an exploitable vulnerability that is beyond Moderate attack potential, this is reported in the ETR as a residual vulnerability.

17.2.4.7.4 Work unit AVA_VAN.4-9

The evaluator ***shall record*** the actual results of the penetration tests.

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 unexpected test results should be investigated. The impact on the evaluation should be stated and justified.

17.2.4.7.5 Work unit AVA_VAN.4-10

The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

Information that would typically be found in the ETR subclause regarding evaluator penetration testing efforts is:

a) TOE test configurations. The particular configurations of the TOE that were penetration tested;

b) TSFI penetration tested. A brief listing of the TSFI and other TOE interfaces that were the focus of the penetration testing;

c) Verdict for the sub-activity. The overall judgement on the results of penetration testing.

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.

17.2.4.7.6 Work unit AVA_VAN.4-11

The evaluator ***shall examine*** the results of all penetration testing to determine that the TOE, in its operational environment, is resistant to an attacker possessing a Moderate attack potential.

If the results reveal that the TOE, in its operational environment, has vulnerabilities exploitable by an attacker possessing less than a High attack potential, then this evaluator action fails.

The guidance in B.2 should be used to determine the attack potential required to exploit a particular vulnerability and whether it can therefore be exploited in the intended environment. It may not be necessary for the attack potential to be calculated in every instance, only if there is some doubt as to whether or not the vulnerability can be exploited by an attacker possessing an attack potential less than high.

17.2.4.7.7 Work unit AVA_VAN.4-12

The evaluator ***shall report*** in the ETR all exploitable vulnerabilities and residual vulnerabilities, detailing for each:

a) its source (e.g. evaluation methodology activity being undertaken when it was conceived, known to the evaluator, read in a publication);

b) the SFR(s) not met;

c) a description;

d) whether it is exploitable in its operational environment or not (i.e. exploitable or residual).

the amount of time, level of expertise, level of knowledge of the TOE, level of opportunity and the equipment required to perform the identified vulnerabilities, and the corresponding values using Tables B.2 and B.3 of B.2.

17.2.5 Evaluation of sub-activity (AVA_VAN.5)

The work units for the evaluation of the sub-activity AVA_VAN.5 are copied from the work units of AVA_VAN.4 as far as possible except that the TOE is attacked by attackers possessing High attack potential.

17.2.5.1 Objectives

The objective of this sub-activity is to determine whether the TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing **High** attack potential.

17.2.5.2 Input

The evaluation evidence for this sub-activity is:

a) the ST;

b) the functional specification;

c) the TOE design;

d) the security architecture description;

e) the implementation representation;

f) the guidance documentation;

g) the TOE suitable for testing;

h) information publicly available to support the identification of possible potential vulnerabilities;

i) the results of the testing of the basic design.

The remaining implicit evaluation evidence for this sub-activity depends on the components that have been included in the assurance package. The evidence provided for each component is to be used as input in this sub-activity.

Other input for this sub-activity is:

a) current information regarding public domain potential vulnerabilities and attacks (e.g. from an evaluation authority).

17.2.5.3 Application notes

The methodical analysis approach takes the form of a structured examination of the evidence. This method requires the evaluator to specify the structure and form the analysis will take (i.e. the manner in which the analysis is performed is predetermined, unlike the focused analysis). The method is specified in terms of the information that will be considered and how/why it will be considered. Further guidance on methodical vulnerability analysis can be found in B.2.2.2.3.

17.2.5.4 Action AVA_VAN.5.1E

17.2.5.4.1 General

CC Part 3 AVA_VAN.5.1C: *The TOE shall be suitable for testing.*

CC Part 3 AVA_VAN.5.2C: *The list of third party components shall include components provided by third parties, and that are part of the TOE or otherwise part of the TOE delivery.*

17.2.5.4.2 Work unit AVA_VAN.5-1

The evaluator ***shall examine*** the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

The TOE provided by the developer and identified in the test plan should have the same unique reference as established by the CM capabilities (ALC_CMC) sub-activities and identified in the ST introduction.

It is possible for the ST to specify more than one configuration for evaluation. The TOE may comprise a number of distinct hardware and software entities that need to be tested in accordance with the ST. The evaluator verifies that all test configurations are consistent with the ST.

The evaluator should consider the security objectives for the operational environment described in the ST that may apply to the test environment and ensure they are met in the testing environment. There may be some objectives for the operational environment that do not apply to the test environment. For example, an objective about user clearances may not apply; however, an objective about a single point of connection to a network would apply.

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.

17.2.5.4.3 Work unit AVA_VAN.5-2

The evaluator ***shall examine*** the TOE to determine that it has been installed properly and is in a known state

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 Evaluation of sub-activity (AGD_PRE.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 and start up the TOE, using the supplied guidance only.

If the evaluator has to perform the installation procedures because the TOE is in an unknown state, this work unit when successfully completed can satisfy work unit AGD_PRE.1-3.

17.2.5.5 Action AVA_VAN.5.2E

17.2.5.5.1 Work unit AVA_VAN.5-3

The evaluator ***shall examine*** sources of information publicly available to identify potential vulnerabilities in the TOE.

The evaluator examines the sources of information publicly available to support the identification of possible potential vulnerabilities in the TOE. There are many sources of publicly available information which the evaluator should consider using items such as those available on the world wide web, including:

a) specialist publications (magazines, books);

b) research papers;

c) conference proceedings.

The evaluator should not constrain their consideration of publicly available information to the above but should consider any other relevant information available.

While examining the evidence provided the evaluator will use the information in the public domain to further search for potential vulnerabilities. Where the evaluators have identified areas of concern, the evaluator should consider information publicly available that relate to those areas of concern.

The availability of information that may be readily available to an attacker that helps to identify and facilitate attacks may substantially enhance the attack potential of a given attacker. The accessibility of vulnerability information and sophisticated attack tools on the Internet makes it more likely that this information will be used in attempts to identify potential vulnerabilities in the TOE and exploit them. Modern search tools make such information easily available to the evaluator, and the determination of resistance to published potential vulnerabilities and well-known generic attacks can be achieved in a cost-effective manner.

The search of the information publicly available should be focused on those sources that refer to the technologies used in the development of the product from which the TOE is derived. The extensiveness of this search should consider the following factors: TOE type, evaluator experience in this TOE type, expected attack potential and the level of ADV evidence available.

The identification process is iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

The evaluator will describe the approach to be taken to identify potential vulnerabilities in the publicly available material, detailing the search to be performed. This may be driven by factors such as areas of concern identified by the evaluator, linked to the evidence the attacker is assumed to be able to obtain. However, it is recognised that in this type of search the approach may further evolve as a result of findings during the search. Therefore, the evaluator will also report any actions taken in addition to those described in the approach to further investigate issues thought to lead to potential vulnerabilities, and will report the evidence examined in completing the search for potential vulnerabilities.

17.2.5.6 Action AVA_VAN.5.3E

17.2.5.6.1 Work unit AVA_VAN.5-4

The evaluator ***shall conduct*** a methodical analysis of ST, guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify possible potential vulnerabilities in the TOE.

Guidance on methodical vulnerability analysis is provided in B.2.2.2.3.

This approach to identification of potential vulnerabilities is to take an ordered and planned approach. A system is to be applied in the examination. The evaluator is to describe the method to be used in terms of the manner in which this information is to be considered and the hypothesis that is to be created.

A flaw hypothesis methodology should be used whereby the ST, development (functional specification, TOE design and implementation representation) and guidance evidence are analysed and then vulnerabilities in the TOE are hypothesised, or speculated.

The evaluator should use the knowledge of the TOE design and operation gained from the TOE deliverables to conduct a flaw hypothesis to identify potential flaws in the development of the TOE and potential errors in the specified method of operation of the TOE.

The security architecture description provides the developer vulnerability analysis, as it documents how the TSF protects itself from interference from untrusted subjects and prevents the bypass of security enforcement functionality. Therefore, the evaluator should build upon the understanding of the TSF protection gained from the analysis of this evidence and then develop this in the knowledge gained from other development (e.g. ADV) evidence.

The approach taken to the methodical search for vulnerabilities is to consider any areas of concern identified in the results of the evaluator's assessment of the development and guidance evidence. However, the evaluator should also consider each aspect of the security architecture

analysis to search for any ways in which the protection of the TSF can be undermined. It may be helpful to structure the methodical analysis on the basis of the material presented in the security architecture description, introducing concerns from other ADV evidence as appropriate. The analysis can then be further developed to ensure all other material from the ADV evidence is considered.

The following provide some examples of hypotheses that may be created when examining the evidence:

a) consideration of malformed input for interfaces available to an attacker at the external interfaces;

b) examination of a key security mechanism cited in the security architecture description, such as process separation, hypothesising internal buffer overflows that may lead to degradation of separation;

c) search to identify any objects created in the TOE implementation representation that are then not fully controlled by the TSF, and can be used by an attacker to undermine SFRs.

For example, the evaluator may identify that interfaces are a potential area of weakness in the TOE and specify an approach to the search that 'all interface specifications in the evidence provided will be searched to hypothesise potential vulnerabilities' and go on to explain the methods used in the hypothesis.

In addition, areas of concern the evaluator has identified during examination of the evidence during the conduct of evaluation activities. Areas of concern may also be identified during the conduct of other work units associated with this component, in particular AVA_VAN.5-7, AVA_VAN.5-5 and AVA_VAN.5-6) where the development and conduct of penetration tests may identify further areas of concerns for investigation, or potential vulnerabilities.

However, examination of only a subset of the development and guidance evidence or their contents is not permitted in this level of rigour. The approach description should provide a demonstration that the methodical approach used is complete, providing confidence that the approach used to search the deliverables has considered all of the information provided in those deliverables.

This approach to identification of potential vulnerabilities is to take an ordered and planned approach; applying a system to the examination. The evaluator is to describe the method to be used in terms of how the evidence will be considered; the manner in which this information is to be considered and the hypothesis that is to be created. This approach should be agreed with the evaluation authority, and the evaluation authority should provide detail of any additional approaches the evaluator should take to the vulnerability analysis and identify any additional information that should be considered by the evaluator.

Although a system to identifying potential vulnerabilities is predefined, the identification process may still be iterative, where the identification of one potential vulnerability may lead to identifying another area of concern that requires further investigation.

Subject to the SFRs the TOE is to meet in the operational environment, the evaluator's independent vulnerability analysis should consider generic potential vulnerabilities under each of the following headings:

a) generic potential vulnerabilities relevant for the type of TOE being evaluated, as may be supplied by the evaluation authority;

b) bypassing;

c) tampering;

d) direct attacks;

e) monitoring;

f) misuse.

Items b) to f) are explained in greater detail in B.2.1.-consideration if the evaluator determines that the potential vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the TOE in its operational environment, which can be used as an input into penetration testing activities, shall be reported in the ETR by the evaluators.

17.2.5.7 Action AVA_VAN.5.4E

17.2.5.7.1 Work unit AVA_VAN.5-6

The evaluator ***shall devise*** penetration tests, based on the independent search for potential vulnerabilities.

The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, to the potential vulnerabilities identified during the search of the sources of publicly available information and the analysis of the TOE guidance and design evidence. The evaluator should have access to current information (e.g. from the evaluation authority) regarding known potential vulnerabilities that may not have been considered by the evaluator.

The evaluator is reminded that, as for considering the security architecture description in the search for vulnerabilities (as detailed in AVA_VAN.5-3), testing should be performed to confirm the architectural properties. If requirements from ATE_DPT are included in the SARs, the developer testing evidence will include testing performed to confirm the correct implementation of any specific mechanisms detailed in the security architecture description. However, the developer testing will not necessarily include testing of all aspects of the architectural properties that protect the TSF, as much of this testing will be negative testing in nature, attempting to disprove the properties. In developing the strategy for penetration testing, the evaluator will ensure that all aspects of the security architecture description are tested, either in functional testing or evaluator penetration testing.

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 potential vulnerability.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a **High** attack potential. 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 an exploitable vulnerability that is beyond **High** attack potential, this is reported in the ETR as a residual vulnerability.

Guidance on determining the necessary attack potential to exploit a potential vulnerability can be found in B.4.

Potential vulnerabilities hypothesised as exploitable by an attacker possessing a **High** (or less) 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.

17.2.5.7.2 Work unit AVA_VAN.5-7

The evaluator ***shall produce*** penetration test documentation for the tests based on the list of potential vulnerabilities in sufficient detail to enable the tests to be repeatable. The test documentation shall include:

a) identification of the potential 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.

The evaluator prepares for penetration testing based on the list of potential vulnerabilities identified during the search of the public domain and the analysis of the evaluation evidence.

The evaluator is not expected to determine the exploitability for potential vulnerabilities beyond those for which a **High** attack potential is required to effect an attack. However, as a result of evaluation expertise, the evaluator may discover a potential vulnerability that is exploitable only by an attacker with greater than **High** attack potential. Such vulnerabilities are to be reported in the ETR as residual vulnerabilities.

With an understanding of the potential vulnerability, the evaluator determines the most feasible way to test for the TOE's susceptibility. Specifically, the evaluator considers:

- the TSFI or other TOE interface that will be used to stimulate the TSF and observe responses (It is possible that the evaluator will need to use an interface to the TOE other than the TSFI to demonstrate properties of the TSF such as those described in the security architecture description (as required by ADV_ARC). It should the noted, that although these TOE interfaces provide a means of testing the TSF properties, they are not the subject of the test.);

- 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);

- special test equipment that will be required to either stimulate a TSFI or make observations of a TSFI;

- whether theoretical analysis should replace physical testing, particularly relevant where the results of an initial test can be extrapolated to demonstrate that repeated attempts of an attack are likely to succeed after a given number of attempts.

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 potential vulnerability.

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.

17.2.5.7.3 Work unit AVA_VAN.5-8

The evaluator ***shall conduct*** penetration testing.

The evaluator uses the penetration test documentation resulting from work unit AVA_VAN.5-6 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 learnt 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.

Should penetration testing show that a hypothesised potential 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.

The evaluator is not expected to test for potential vulnerabilities (including those in the public domain) beyond those which required a **High** attack potential. 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 an exploitable vulnerability that is beyond **High** attack potential, this is reported in the ETR as a residual vulnerability.

17.2.5.7.4 Work unit AVA_VAN.5-9

The evaluator ***shall record*** the actual results of the penetration tests.

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 unexpected test results should be investigated. The impact on the evaluation should be stated and justified.

17.2.5.7.5 Work unit AVA_VAN.5-10

The evaluator ***shall report*** in the ETR the evaluator penetration testing effort, outlining the testing approach, configuration, depth and results.

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 evaluation authorities 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.

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) TSFI penetration tested. A brief listing of the TSFI and other TOE interfaces that were the focus of the penetration testing;

c) Verdict for the sub-activity. The overall judgement on the results of penetration testing.

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.

17.2.5.7.6 Work unit AVA_VAN.5-11

The evaluator ***shall examine*** the results of all penetration testing to determine that the TOE, in its operational environment, is resistant to an attacker possessing a **High** attack potential.

If the results reveal that the TOE, in its operational environment, has vulnerabilities exploitable by an attacker possessing an attack potential less than **or equal to** High, then this evaluator action fails.

The guidance in B.4 and the guidance for special technical areas that is relevant for the national scheme should be used to determine the attack potential required to exploit a particular vulnerability and whether it can therefore be exploited in the intended environment. It may not be necessary for the attack potential to be calculated in every instance, only if there is some doubt as to whether or not the vulnerability can be exploited by an attacker possessing an attack potential less than **or equal to** High.

17.2.5.7.7 Work unit AVA_VAN.5-12

The evaluator ***shall report*** in the corresponding ETR-part 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 SFR(s) not met;

c) a description;

d) whether it is exploitable in its operational environment or not (i.e. exploitable or residual);

e) the amount of time, level of expertise, level of knowledge of the TOE, level of opportunity and the equipment required to perform the identified vulnerabilities, and the corresponding values using Tables B.2 and B.3 of Annex B.

17.3 Composite vulnerability assessment (AVA_COMP)

17.3.1 General

The composite-specific work units defined here are intended to be integrated as refinements to the evaluation activities of the AVA class listed in the following table. The other activities of the AVA class do not require composite-specific work units.

Table 5 — AVA_COMP

CC assurance family	Evaluation activity	Evaluation work unit	Composite-specific work unit
AVA_VAN	AVA_VAN.1.3E	AVA_VAN.1-5	AVA_COMP.1-1
	AVA_VAN.1.3E	AVA_VAN.1-6	AVA_COMP.1-2
	AVA_VAN.1.3E	AVA_VAN.1-7	AVA_COMP.1-2
	AVA_VAN.1.3E	AVA_VAN.1-8	AVA_COMP.1-2

NOTE If the level of the assurance requirement chosen is higher than those identified in this table, the composite-specific work unit is also applicable.

17.3.2 Evaluation of sub-activity (AVA_COMP.1)

17.3.2.1 Objectives

The aim of this activity is to determine the exploitability of flaws or weaknesses in the composite product as a whole in the intended environment.

17.3.2.2 Application notes

This activity focuses exclusively on the vulnerability assessment of the composite product as a whole and represents merely partial efforts within the general approach being covered by the standard assurance family of the class AVA: AVA_VAN.

The composite product evaluator shall perform a vulnerability analysis for the composite product using, among others, the results of the base component evaluation. This vulnerability analysis shall be confirmed by penetration testing.

The composite product evaluator shall check that the confidentiality protection of the dependent component embedded into/installed onto the base component is consistent with the confidentiality level claimed by the dependent component developer for ALC_DVS.

In special cases, the vulnerability analysis and the definition of attacks can be difficult, need considerable time and require extensive pre-testing, if only documentation is available. The base component may also be used in a way that was not foreseen by the base component developer and the base component evaluator, or the dependent component developer may not have followed the (security) requirements provided with the base component and its evaluation-related documentation. Different possibilities exist to shorten composite product vulnerability analysis in such cases. For example, the composite product evaluator may consult the base component evaluator and draw on his experience gained during the base component evaluation. Alternatively, an approach aiming at the separation of vulnerabilities of the dependent component and the base component by using specific test samples of the base component on which the composite product evaluator may load test dependent components at his own discretion. The intention hereby is to use test dependent components without countermeasures and without deactivating any base component inherent countermeasure.

Within the composite evaluation technique, the so-called *ETR for composite evaluation* and the evaluation authority report play an important role. Please refer to CC Part 1, 14.3.3.6 and 14.3.3.7 for more details, in particular concerning objective and contents of these documents and corresponding rules for their handling and usage in the framework of the composite evaluation approach.

The results of the vulnerability assessment for the base component of the composite product represented in the *ETR for composite evaluation* can be reused under the following conditions: they are up-to-date and all composite evaluation activities for correctness -- ASE_COMP.1, ALC_COMP.1, ADV_COMP.1 and ATE_COMP.1 -- were finalised with the verdict PASS. The validity and topicality of the base component's report of the base component evaluation authority and of the *ETR for composite evaluation* are a prerequisite for using such documents in the composite evaluation approach and therefore shall be verified by the composite product evaluator.

Due to composing of the base component and the dependent component additional vulnerabilities of the base component can occur that might be not mentioned or addressed in the *ETR for composite evaluation*. In these circumstances the composite product evaluation authority may require a re-assessment or re-evaluation of the base component focusing on the new vulnerabilities' issues.

The composite product evaluation sponsor shall ensure that the following is made available for the composite product evaluator:

- the base component-related user guidance;
- the base component-related *ETR for composite evaluation* prepared by the base component evaluator;
- the base component's report of the base component evaluation authority.

17.3.2.3 Action AVA_COMP.1.1E

17.3.2.3.1 General

CC Part 3 AVA_COMP.1.1C: *The composite product provided shall be suitable for testing as a whole.*

17.3.2.3.2 Work unit AVA_COMP.1-1

The evaluator ***shall examine*** the results of the vulnerability assessment for the base component of the composite product to determine that they can be re-used for the composite evaluation.

The results of the vulnerability assessment for the base component of the composite product are usually represented in the *ETR for composite evaluation*. They can be re-used if the following conditions are met: they are up-to-date and all composite evaluation activities for correctness -- ASE_COMP.1, ALC_COMP.1, ADV_COMP.1 and ATE_COMP.1 -- resulted in verdicts of PASS.

The validity and topicality of the base component's report of the base component evaluation authority and of the *ETR for composite evaluation* are a prerequisite for their re-use in the composite evaluation of the composite product and shall be verified by the composite product evaluator.

It is noted that the base component itself can be a composite product. This means also that the validity and topicality of each evaluation authority report and *ETR for composite evaluation* of all the products / TOEs that compose the base component shall be checked.

When the relevance of the *ETR(s) for composite evaluation* for the composite product is checked, the necessity of checking their contents depends on the dependent component of the composite product and the user available TSFI. If the base component-TSFI are available to the user of the composite product or are used by the dependent component, the contents of the *ETR(s) for composite evaluation* shall be checked. Otherwise, such relevance check of contents might be not necessary.

The evaluator shall also consider the relevant requirements in any base component-related report of the base component evaluation authority.

The result of this work unit shall be integrated to the result of AVA_VAN.1.3E / AVA_VAN.1-5 (or the equivalent higher components if a higher assurance level is selected).

17.3.2.3.3 Work unit AVA_COMP.1-2

The evaluator ***shall specify, conduct and document*** penetration testing of the composite product as a whole, using the standard approach of the assurance family AVA_VAN.

If all correctness-related activities (ASE_COMP.1, ALC_COMP.1, ADV_COMP.1 and ATE_COMP.1) have resulted in verdicts of PASS, and the composite product's base component with its base component evaluation covers all security properties needed for the composite product, then, a composition of this base component and the dependent component may also not create additional vulnerabilities of the base component.

If the evaluator determined that composing of the base component and the dependent component creates additional vulnerabilities of the base component, a contradiction to the verdict PASS for the correctness activities has to be supposed or the report of the base component evaluation authority for the base component does not cover all security properties and evaluation-related information of the base component as needed for the current composite product and its evaluation.

The result of this work unit shall be integrated to the result of AVA_VAN.1.3E / AVA_VAN.1-6, AVA_VAN.1-7, AVA_VAN.1-8 (or the equivalent higher components if a higher assurance level is selected).

18 Class ACO: Composition

18.1 General

The goal of this activity is to determine whether the components can be integrated in a secure manner, as defined in the ST for the composed TOE. This is achieved through examination and testing of the interfaces between the components, supported by examination of the design of the components and the conduct of vulnerability analysis.

18.2 Application notes

The Reliance of dependent component (ACO_REL) family identifies where the dependent component is reliant upon IT in its operational environment (satisfied by a base component in the composed TOE evaluation) in order to provide its own security services. This reliance is identified in terms of the interfaces expected by the dependent component to be provided by the base component. Development evidence (ACO_DEV) then determines which interfaces of the base component were considered (as TSFI) during the component evaluation of the base component.

It should be noted that Reliance of dependent component (ACO_REL) does not cover other evidence that may be needed to address the technical integration problem of composing components (e.g. descriptions of non-TSF interfaces of the operating system, rules for integration, etc.). This is outside the security assessment of the composition and is a functional composition issue.

As part of Composed TOE testing (ACO_CTT) the evaluator will perform testing of the composed TOE SFRs at the composed TOE interfaces and of the interfaces of the base component relied upon by the dependent component to confirm they operate as specified. The subset selected will consider the possible effects of changes to the configuration/use of the base component as used in the composed TOE. These changes are identified from the configuration of the base component determined during the base component evaluation. The developer will provide test evidence for each of the base component interfaces (the requirements for coverage are consistent with those applied to the evaluation of the base component).

Composition rationale (ACO_COR) requires the evaluator to determine whether the appropriate assurance measures have been applied to the base component, and whether the base component is being used in its evaluated configuration. This includes determination of whether all security functionality required by the dependent component was within the TSF of the base component. The Composition rationale (ACO_COR) requirement may be met through the production of evidence that each of these is demonstrated to be upheld. This evidence may be in the form of the security target and a public report of the component evaluation (e.g. certification report).

If, on the other hand, one of the above have not been upheld, then it may be possible that an argument can be made as to why the assurance gained during an original evaluation is unaffected. If this is not possible then additional evaluation evidence for those aspects of the base component not covered may have to be provided. This material is then assessed in Development evidence (ACO_DEV).

For example, it may be the case as described in the Interactions between entities (see B.3, Interactions between composed IT entities in CC Part 3) that the dependent component requires the base component to provide more security functionality in the composed TOE than included in the base component evaluation. This would be determined during the application of the Reliance of dependent component (ACO_REL) and Development evidence (ACO_DEV) families. In this case the composition rationale evidence provided for Composition rationale (ACO_COR) would demonstrate that the assurance gained from the base component evaluation is unaffected. This may be achieved by means including:

Performing a re-evaluation of the base component focusing on the evidence relating to the extended part of the TSF;

Demonstrating that the extended part of the TSF cannot affect other portions of the TSF, and providing evidence that the extended part of the TSF provides the necessary security functionality.

18.3 Composition rationale (ACO_COR)

18.3.1 Evaluation of sub-activity (ACO_COR.1)

18.3.1.1 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the composition rationale;

c) the reliance information;

d) the development information;

e) unique identifier.

18.3.1.2 Action ACO_COR.1.1E

18.3.1.2.1 General

CC Part 3 ACO_COR.1.1C: *The composition rationale shall demonstrate that a level of assurance at least as high as that of the dependent component has been obtained for the support functionality of the base component, when the base component is configured as required to support the TSF of the dependent component.*

18.3.1.2.2 Work unit ACO_COR.1-1

The evaluator ***shall examine*** the correspondence analysis with the development information and the reliance information to identify the interfaces that are relied upon by the dependent component which are not detailed in the development information.

The evaluator's goal in this work unit is two-fold:

- to determine which interfaces relied upon by the dependent component have had the appropriate assurance measures applied;

- to determine that the assurance package applied to the base component during the base component evaluation contained either the same assurance requirements as those in the package applied to the dependent component during its' evaluation, or hierarchically higher assurance requirements.

The evaluator may use the correspondence tracing in the development information developed during the Development evidence (ACO_DEV) activities (e.g. ACO_DEV.1-2**,** ACO_DEV.2-4**,** ACO_DEV.3-6) to help identify the interfaces identified in the reliance information that are not considered in the development information.

The evaluator will record the SFR-enforcing interfaces described in the reliance information that are not included in the development information. These will provide input to ACO_COR.1-3 work unit, helping to identify the portions of the base component in which further assurance is required.

If both the base and dependent components were evaluated against the same assurance package, then the determination of whether the level of assurance in the portions within the base component evaluation is at least as high as that of the dependent component is trivial. If however, the assurance packages applied to the components during the component evaluations differ, the evaluator needs to determine that the assurance requirements applied to the base component are all hierarchically higher to the assurance requirements applied to the dependent component.

18.3.1.2.3 Work unit ACO_COR.1-2

The evaluator ***shall examine*** the composition rationale to determine, for those included base component interfaces on which the dependent TSF relies, whether the interface was considered during the evaluation of the base component.

The ST, component public evaluation report (e.g. certification report) and guidance documents for the base component all provide information on the scope and boundary of the base component. The ST provides details of the logical scope and boundary of the composed TOE, allowing the evaluator to determine whether an interface relates to a portion of the product that was within the scope of the evaluation. The guidance documentation provides details of use of all interfaces for the composed TOE. Although the guidance documentation may include details of interfaces in the product that are not within the scope of the evaluation, any such interfaces should be identifiable, either from the scoping information in the ST or through a portion of the guidance that deals with the evaluated configuration. The public evaluation report may provide any additional constraints on the use of the composed TOE that are necessary.

Therefore, the combination of these inputs allows the evaluator to determine whether an interface described in the composition rationale has the necessary assurance associated with it, or whether further assurance is required. The evaluator will record those interfaces of the base component for which additional assurance is required, for consideration during ACO_COR.1-3.

18.3.1.2.4 Work unit ACO_COR.1-3

The evaluator ***shall examine*** the composition rationale to determine that the necessary assurance measures have been applied to the base component.

The evaluation verdicts, and resultant assurance, for the base component can be reused provided the same portions of the base component are used in the composed TOE and they are used in a consistent manner.

In order to determine whether the necessary assurance measures have already been applied to the component, and the portions of the component for which assurance measures still need to be applied, the evaluator should use the output of the ACO_DEV.*.2E action and the work units ACO_COR.1-1 and ACO_COR.1-2**.**

For those interfaces identified in the reliance information [Reliance of dependent component (ACO_REL)], but not discussed in development information [Development evidence (ACO_DEV)]**,** additional information is required. (Identified in ACO_COR.1-1.)

For those interfaces used inconsistently in the composed TOE from the base component (difference between the information provided in Development evidence (ACO_DEV) and Reliance of dependent component (ACO_REL) the impact of the differences in use need to be considered. (Identified in ACO_DEV.*.2E.)

For those interfaces identified in composition rationale for which no assurance has previously been gained, additional information is required. (Identified in ACO_COR.1-2.)

For those interfaces consistently described in the reliance information, composition rationale and the development information, no further action is required as the results from the base component evaluation can be re-used.

The interfaces of the base component reported to be required by the reliance information but not included in the development information indicate the portions of the base component where further assurance is required. The interfaces identify the entry points into the base component.

For those interfaces included in both the development information and reliance information, the evaluator is to determine whether the interfaces are being used in the composed TOE in a manner that is consistent with the base component evaluation. The method of use of the interface will be considered during the Development evidence (ACO_DEV) activities to determine that the use of the interface is consistent in both the base component and the composed TOE. The remaining consideration is the determination of whether the configurations of the base component and the composed TOE are consistent. To determine this, the evaluator will consider the guidance documentation of each to ensure they are consistent (see further guidance below regarding consistent guidance documentation). Any deviation in the documentation will be further analysed by the evaluation to determine the possible effects.

For those interfaces that are consistently described in the reliance information and development information, and for which the guidance is consistent for the base component and the composed TOE, the required level of assurance has been provided.

Table 5 provides guidance on how to determine consistency between assurance gained in the base component, the evidence provided for the composed TOE, and the analysis performed by the evaluator in the instances where inconsistencies are identified.

Table 5 - Guidance on how to determine consistency

a)	**Development**	The reliance information identifies the interfaces in the dependent component that are to be matched by the base component. If an interface identified in the reliance information is not identified in the development information, then the composition rationale is to provide a justification of how the base component provides the required interfaces. If an interface identified in the reliance information is identified in the development information, but there are inconsistencies between the descriptions, further analysis is required. The evaluator identifies the differences in use of the base component as considered in the base component evaluation and the composed TOE evaluation. The evaluator will devise testing to be performed (during the conduct of Composed TOE testing (ACO_CTT)) to test the interface. The patch status of the base and dependent components as used in the composed TOE should be compared to the patch status of the components during the component evaluations. If any patches have been applied to the components, the composition rationale is to include details of the patches, including any potential impact to the SFRs of the evaluated component. The evaluator should consider the details of the changes provided and verify the accuracy of the potential impact of the change on the component SFRs. The evaluator should then consider whether the changes made by the patch should be verified

		through testing, and will identify the necessary testing approach. The testing may take the form of repeating the applicable evaluator/developer testing performed for the component evaluation of the component or it may be necessary for the evaluator to devise new tests to confirm the modified component. If any of the individual components have been the subject of assurance continuity activities since the completion of the component evaluation, the evaluator will consider the changes assessed in the assurance continuity activities during the independent vulnerability analysis activity for the composed TOE (in Composition vulnerability analysis (ACO_VUL)).
b)	**Guidance**	The guidance for the composed TOE is likely to make substantial reference out to the guidance for the individual components. The minimal guidance expected to be necessary is the identification of any ordering dependencies in the application of guidance for the dependent and base components, particularly during the preparation (installation) of the composed TOE. In addition to the application of the Preparative procedures (AGD_PRE) and Operational user guidance (AGD_OPE) families to the guidance for the composed TOE, it is necessary to analyse the consistency between the guidance for the components and the composed TOE, to identify any deviations. If the composed TOE guidance refers out to the base component and dependent component guidance, then the consideration for consistency is limited to consistency between the guidance documentation provided for each of the components (i.e. consistency between the base component guidance and the dependent component guidance). However, if additional guidance is provided for the composed TOE, to that provided for the components, greater analysis is required, as consistency is also required between the guidance documentation for the components and guidance documentation for the composed TOE. Consistent in this instance is understood to mean that either the guidance is the same or it places additional constraints on the operation of the individual components when combined, in a similar manner to refinement of functional/assurance components. With the information available (that used as input for Development evidence (ACO_DEV) or the development aspects discussed above) the evaluator may be able to determine all possible impacts of the deviation from the configuration of the base component specified in the component evaluation. However, for high EALs (where evaluation of the base component included requirements) it is possible that, unless detailed design abstractions for the base component are delivered as part of the development information for the composed TOE, the possible impacts of the modification to the

		guidance cannot be fully determined as the internals are unknown. In this case the evaluator will report the residual risk of the analysis. These residual risks are to be included in any public evaluation report for the composed TOE. The evaluator will note these variances in the guidance for input into evaluator independent testing activities (Composed TOE testing (ACO_CTT)). The guidance for the composed TOE may add to the guidance for the components, particularly in terms of installation and the ordering of installation steps for the base component in relation to the installation steps for the dependent component. The ordering of the steps for the installation of the individual components should not change, however they may need to be interleaved. The evaluator will examine this guidance to ensure that it still meets the requirement of the AGD_PRE activity performed during the evaluations of the components. It may be the case that the reliance information identifies that interfaces of the base component, in addition to those identified as TSFIs of the base component, are relied upon by the dependent component are identified in the reliance information. It may be necessary for guidance to be provided for the use of any such additional interfaces in the base component. Provided the consumer of the composed TOE is to receive the guidance documentation for the base component, then the results of the AGD_PRE and AGD_OPE verdicts for the base component can be reused for those interfaces considered in the evaluation of the base component. However, for the additional interfaces relied upon by the dependent component, the evaluator will need to determine that the guidance documentation for the base component meets the requirements of AGD_PRE and AGD_OPE, as applied in the base component evaluations. For those interfaces considered during the base component evaluation, and therefore, for which assurance has already been gained, the evaluator will ensure that the guidance for the use of each interface for the composed TOE is consistent with that provided for the base component. To determine the guidance for the composed TOE is consistent with that for the base component, the evaluator should perform a mapping for each interface to the guidance provided for both the composed TOE and the base component. The evaluator then compares the guidance to determine consistency. Examples of additional constraints provided in composed TOE guidance that would be considered to be consistent with component guidance are (guidance for a component is given followed by an example of guidance for a composed TOE that would be considered to provide additional constraints):

		• Component: The password length must be set to a minimum of 8 characters length, including alphabetic and numeric characters. • Composed TOE: The password length must be set to a minimum of 10 characters in length, including alphabetic and numeric characters and at least one of the following special characters: ( ) { } ^ < > - _ • NOTE: It would only be acceptable to increase the password length to [integer > 8] characters while removing the mandate for the inclusion of both alphabetic and numeric characters for the composed TOE, if the same or a higher metric was achieved for the strength rating (taking into account the likelihood of the password being guessed). • Component: The following services are to be disabled in the registry settings: WWW Publishing Service and ICDBReporter service. • Composed TOE: The following services are to be disabled in the registry settings: Publishing Service, ICDBReporter service, Remote Procedure Call (RPC) Locator and Procedure Call (RPC) Service. • Component: Select the following attributes to be included in the accounting log files: date, time, type of event, subject identity and success/failure. • Composed TOE: Select the following attributes to be included in the accounting log files: date, time, type of event, subject identity, success/failure, event message and process thread. If the guidance for the composed TOE deviates (is not a refinement) from that provided for the base component, the evaluator will assess the potential risks of the modification to the guidance. The evaluator will use the information available (including that provided in the public domain, the architectural description of the base component in the public evaluation report (e.g. certification report), the context of the guidance from the remainder of the guidance documentation) to identify likely impact of the modification to the guidance on the SFRs of the composed TOE. If during the dependent component evaluation the trial installation used the base component to satisfy the environment requirements of the dependent component this work unit for the composed TOE is considered to be satisfied. If the base component was not used in satisfaction of the work unit AGD_PRE.1-3 during the dependent component evaluation, the evaluator will apply the user procedures provided for the composed TOE to prepare the composed TOE, in accordance with the guidance specified in AGD_PRE.1-3. This will allow the evaluator to determine that the preparative guidance provided for the composed TOE is sufficient to prepare the composed TOE and its operational environment securely.

c)	**Delivery**	If there is a different delivery mechanism used for the delivery of the composed TOE (i.e. the components are not delivered to the consumer in accordance with the secure delivery procedures defined and assessed during the evaluation of the components), the delivery procedures for the composed TOE will require evaluation against the Delivery (ALC_DEL) requirements applied during the components evaluations. The composed TOE may be delivered as an integrated product or may require the components to be delivered separately. If the components are delivered separately, the results of the delivery of the base component and dependent component are reused. The delivery of the base component is checked during the evaluator trial installation of the dependent component, using the specified guidance and checking the aspects of delivery that are the responsibility of the user, as described in the guidance documentation for the base component. If the composed TOE is delivered as a new entity, then the method of delivery of that entity must be considered in the composed TOE evaluation activities. The assessment of the delivery procedures for composed TOE items is to be performed in accordance with the methodology for Delivery (ALC_DEL) as for any other [component] TOE, ensuring any additional items (e.g. additional guidance documents for the composed TOE) are considered in the delivery procedures.
d)	**CM Capabilities**	The unique identification of the composed TOE is considered during the application of Evaluation of sub-activity (ALC_CMC.1) and the items from which that composed TOE is comprised are considered during the application of Evaluation of sub-activity (ALC_CMS.2). Although additional guidance may be produced for the composed TOE, the unique identification of this guidance (considered as part of the unique identification of the composed TOE during Evaluation of sub-activity (ALC_CMC.1)) is considered sufficient control of the guidance. The verdicts of the remaining (not considered above) Class ALC: Life-cycle support activities can be reused from the base component evaluation, as no further development is performed during integration of the composed TOE. There are no additional considerations for development security as the integration is assumed to take place at either the consumer's site or, in the instance that the composed TOE is delivered as an integrated product, at the site of the dependent component developer. Control at the consumer's site is outside the consideration of the CC (all parts). No additional requirements or guidance are necessary if integration is at the same site as that for the dependent component, as all components are considered to be configuration items for the

		composed TOE, and should therefore be considered under the dependent component developer's security procedures anyway. Tools and techniques adopted during integration will be considered in the evidence provided by the dependent component developer. Any tools/techniques relevant to the base component will have been considered during the evaluation of the base component. For example, if the base component is delivered as source code and requires compilation by the consumer (e.g. dependent component developer who is performing integration) the compiler would have been specified and assessed, along with the appropriate arguments, during evaluation of the base component. There is no life-cycle definition applicable to the composed TOE, as no further development of items is taking place. The results of flaw remediation for a component are not applicable to the composed TOE. If flaw remediation is included in the assurance package for the composed TOE, then the Flaw remediation (ALC_FLR) requirements are to be applied during the composed TOE evaluation (as for any augmentation).
e)	**Tests**	The composed TOE will have been tested during the conduct of the Class ATE: Tests activities for evaluation of the dependent component, as the configurations used for testing of the dependent component should have included the base component to satisfy the requirements for IT in the operational environment. If the base component was not used in the testing of the dependent component for the dependent component evaluation, or the configuration of either component varied from their evaluated configurations, then the developer testing performed for evaluation of the dependent component to satisfy the Class ATE: Tests requirements is to be repeated on the composed TOE.

18.4 Development evidence (ACO_DEV)

18.4.1 Evaluation of sub-activity (ACO_DEV.1)

18.4.1.1 Objectives

The objective of this sub-activity is to determine that the appropriate security functionality is provided by the base component to support the dependent component. This is achieved through examination of the interfaces of the base component to determine that they are consistent with the interfaces specified in the reliance information; those required by the dependent component.

The description of the interfaces into the base component is to be provided at a level of detail consistent with Evaluation of sub-activity (ADV_FSP.2) although not all of the aspects necessary for satisfaction of Evaluation of sub-activity (ADV_FSP.2) are required for Evaluation of sub-activity (ACO_DEV.1), as once the interface has been identified and the purpose described the remaining detail of the interface specification can be reused from evaluation of the base component.

18.4.1.2 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the development information;

c) the reliance information.

18.4.1.3 Action ACO_DEV.1.1E

18.4.1.3.1 General

CC Part 3 ACO_DEV.1.1C: *The development information shall describe the purpose of each interface of the base component used in the composed TOE.*

18.4.1.3.2 Work unit ACO_DEV.1-1

The evaluator ***shall examine*** the development information to determine that it describes the purpose of each interface.

The base component provides interfaces to support interaction with the dependent component in the provision of the dependent TSF. The purpose of each interface is to be described at the same level as the description of the interfaces to the dependent component TSF functionality, as would be provided between subsystems in the TOE design [Evaluation of sub-activity (ADV_TDS.1)]**.** This description is to provide the reader with an understanding of how the base component provides the services required by the dependent component TSF.

This work unit may be satisfied by the provision of the functional specification for the base component for those interfaces that are TSFIs of the base component.

CC Part 3 ACO_DEV.1.2C: *The development information shall show correspondence between the interfaces, used in the composed TOE, of the base component and the dependent component to support the TSF of the dependent component.*

18.4.1.3.3 Work unit ACO_DEV.1-2

The evaluator ***shall examine*** the development information to determine the correspondence, between the interfaces of the base component and the interfaces on which the dependent component relies, is accurate.

The correspondence between the interfaces of the base component and the interfaces on which the dependent component relies may take the form of a matrix or table. The interfaces that are relied upon by the dependent component are identified in the reliance information [as examined during Reliance of dependent component (ACO_REL) activity].

There is, during this activity, no requirement to determine completeness of the coverage of interfaces that are relied upon by the dependent component, only that the correspondence is correct and ensuring that interfaces of the base component are mapped to interfaces required by the dependent component wherever possible. The completeness of the coverage is considered in Composition rationale (ACO_COR) activities.

18.4.1.4 Action ACO_DEV.1.2E

18.4.1.4.1 Work unit ACO_DEV.1-3

The evaluator ***shall examine*** the development information and the reliance information to determine that the interfaces are described consistently.

The evaluator's goal in this work unit is to determine that the interfaces described in the development information for the base component and the reliance information for the dependent component are represented consistently.

18.4.2 Evaluation of sub-activity (ACO_DEV.2)

18.4.2.1 Objectives

The objective of this sub-activity is to determine that the appropriate security functionality is provided by the base component to support the dependent component. This is achieved through examination of the interfaces and associated security behaviour of the base component to determine that they are consistent with the interfaces specified in the reliance information; those required by the dependent component.

18.4.2.2 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the development information;

c) reliance information.

18.4.2.3 Action ACO_DEV.2.1E

18.4.2.3.1 General

CC Part 3 ACO_DEV.2.1C: *The development information shall describe the purpose and method of use of each interface of the base component used in the composed TOE.*

18.4.2.3.2 Work unit ACO_DEV.2-1

The evaluator ***shall examine*** the development information to determine that it describes the purpose of each interface.

The base component provides interfaces to support interaction with the dependent component in the provision of the dependent TSF. The purpose of each interface is to be described at the same level as the description of the interfaces to the dependent component TSF functionality, as would be provided between subsystems in the TOE design [Evaluation of sub-activity (ADV_TDS.1)]. This description is to provide the reader with an understanding of how the base component provides the services required by the dependent component TSF.

This work unit may be satisfied by the provision of the functional specification for the base component for those interfaces that are TSFIs of the base component.

18.4.2.3.3 Work unit ACO_DEV.2-2

The evaluator ***shall examine*** the development information to determine that it describes the method of use for each interface.

The method of use for an interface summarizes how the interface is manipulated in order to invoke the operations and obtain results associated with the interface. The evaluator should be able to determine from reading this material in the development information how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each interface, as it may be possible to describe in general how APIs are invoked, for instance, and then identify each interface using that general style.

This work unit may be satisfied by the provision of the functional specification for the base component for those interfaces that are TSFIs of the base component.

CC Part 3 ACO_DEV.2.2C: *The development information shall provide a high-level description of the behaviour of the base component, which supports the enforcement of the dependent component SFRs.*

18.4.2.3.4 Work unit ACO_DEV.2-3

The evaluator ***shall examine*** the development information to determine that it describes the behaviour of the base component that supports the enforcement of the dependent component SFRs.

The dependent component invokes interfaces of the base component for the provision of services by the base component. For the interfaces of the base component that are invoked, the development information shall provide a high-level description of the associated security behaviour of the base component. The description of the base component security behaviour will outline how the base component provides the necessary service when the call to the interface is made. This description is to be at a level similar to that provided for ADV_TDS.1.4C. Therefore, the provision of the TOE design evidence from the base component evaluation would satisfy this work unit, where the interfaces invoked by the dependent component are TSFI of the base component. If the interfaces invoked by the dependent component are not TSFIs of the base component it is the associated security behaviour will not necessarily be described in the base component TOE design evidence.

CC Part 3 ACO_DEV.2.3C: *The development information shall show correspondence between the interfaces, used in the composed TOE, of the base component and the dependent component to support the TSF of the dependent component.*

18.4.2.3.5 Work unit ACO_DEV.2-4

The evaluator ***shall examine*** the development information to determine the correspondence, between the interfaces of the base component and the interfaces on which the dependent component relies, is accurate.

The correspondence between the interfaces of the base component and the interfaces on which the dependent component relies may take the form of a matrix or table. The interfaces that are relied upon by the dependent component are identified in the reliance information [as examined during Reliance of dependent component (ACO_REL)]**.**

There is, during this activity, no requirement to determine completeness of the coverage of interfaces that are relied upon by the dependent component, only that the correspondence is correct and ensuring that interfaces of the base component are mapped to interfaces required by the dependent component wherever possible. The completeness of the coverage is considered in Composition rationale (ACO_COR) activities.

18.4.2.4 Action ACO_DEV.2.2E

18.4.2.4.1 Work unit ACO_DEV.2-5

The evaluator ***shall examine*** the development information and the reliance information to determine that the interfaces are described consistently.

The evaluator's goal in this work unit is to determine that the interfaces described in the development information for the base component and the reliance information for the dependent component are represented consistently.

18.4.3 Evaluation of sub-activity (ACO_DEV.3)

18.4.3.1 Objectives

The objective of this sub-activity is to determine that the appropriate security functionality is provided by the base component to support the dependent component. This is achieved through examination of the interfaces and associated security behaviour of the base component to determine that they are consistent with the interfaces specified in the reliance information; those required by the dependent component.

In addition to the interface description, the subsystems of the base component that provide the security functionality required by the dependent component will be described to enable the evaluator to determine whether or not that interface formed part of the TSF of the base component.

18.4.3.2 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the development information;

c) reliance information.

18.4.3.3 Action ACO_DEV.3.1E

18.4.3.3.1 General

CC Part 3 ACO_DEV.3.1C: *The development information shall describe the purpose and method of use of each interface of the base component used in the composed TOE.*

18.4.3.3.2 Work unit ACO_DEV.3-1

The evaluator ***shall examine*** the development information to determine that it describes the purpose of each interface.

The base component provides interfaces to support interaction with the dependent component in the provision of the dependent TSF. The purpose of each interface is to be described at the same level as the description of the interfaces to the dependent component TSF functionality, as would be provided between subsystems in the TOE design [Evaluation of sub-activity (ADV_TDS.1)]. This description is to provide the reader with an understanding of how the base component provides the services required by the dependent component TSF.

This work unit may be satisfied by the provision of the functional specification for the base component for those interfaces that are TSFIs of the base component.

18.4.3.3.3 Work unit ACO_DEV.3-2

The evaluator ***shall examine*** the development information to determine that it describes the method of use for each interface.

The method of use for an interface summarizes how the interface is manipulated in order to invoke the operations and obtain results associated with the interface. The evaluator should be able to determine from reading this material in the development information how to use each interface. This does not necessarily mean that there needs to be a separate method of use for each interface, as it may be possible to describe in general how APIs are invoked, for instance, and then identify each interface using that general style.

This work unit may be satisfied by the provision of the functional specification for the base component for those interfaces that are TSFIs of the base component.

CC Part 3 ACO_DEV.3.2C: *The development information shall identify the subsystems of the base component that provide interfaces of the base component used in the composed TOE.*

18.4.3.3.4 Work unit ACO_DEV.3-3

The evaluator ***shall examine*** the development information to determine that all subsystems of the base component that provide interfaces to the dependent component are identified.

For those interfaces that are considered to form part of the TSFI of the base component, the subsystems associated with the interface will be subsystems considered in the TOE design (ADV_TDS) activity during the base component evaluation. The interfaces on which the dependent component relies that did not form part of the TSFI of the base component will map to subsystems outside of the base component TSF.

CC Part 3 ACO_DEV.3.3C: *The development information shall provide a high-level description of the behaviour of the base component subsystems, which support the enforcement of the dependent component SFRs.*

18.4.3.3.5 Work unit ACO_DEV.3-4

The evaluator ***shall examine*** the development information to determine that it describes the behaviour of the base component subsystems that support the enforcement of the dependent component SFRs.

The dependent component invokes interfaces of the base component for the provision of services by the base component. For the interfaces of the base component that are invoked, the development information shall provide a high-level description of the associated security behaviour of the base component. The description of the base component security behaviour will outline how the base component provides the necessary service when the call to the interface is made. This description is to be at a level similar to that provided for ADV_TDS.1.4C. Therefore, the provision of the TOE design evidence from the base component evaluation would satisfy this work unit, where the interfaces invoked by the dependent component are TSFI of the base component. If the interfaces invoked by the dependent component are not TSFIs of the base component it is the associated security behaviour will not necessarily be described in the base component TOE design evidence.

CC Part 3 ACO_DEV.3.4C: *The development information shall provide a mapping from the interfaces to the subsystems of the base component.*

18.4.3.3.6 Work unit ACO_DEV.3-5

The evaluator ***shall examine*** the development information to determine that the correspondence between the interfaces and subsystems of the base component is accurate.

If the TOE design and functional specification evidence from the base component evaluation is available, this can be used to verify the accuracy of the correspondence between the interfaces and subsystems of the base component as used in the composed TOE. Those interfaces of the base component, which formed part of the base component TSFI will be described in the base component functional specification, and the associated subsystems will be described in the base component TOE design evidence. The tracing between the two will be provided in the base component TOE design evidence.

If, however, the base component interface did not form part of the TSFI of the base component, the description of the subsystem behaviour provided in the development information will be used to verify the accuracy of the correspondence.

CC Part 3 ACO_DEV.3.5C: *The development information shall show correspondence between the interfaces, used in the composed TOE, of the base component and the dependent component to support the TSF of the dependent component.*

18.4.3.3.7 Work unit ACO_DEV.3-6

The evaluator ***shall examine*** the development information to determine the correspondence, between the interfaces of the base component and the interfaces on which the dependent component relies, is accurate.

The correspondence between the interfaces of the base component and the interfaces on which the dependent component relies may take the form of a matrix or table. The interfaces that are relied upon by the dependent component are identified in the reliance information [as examined during Reliance of dependent component (ACO_REL)].

There is, during this activity, no requirement to determine completeness of the coverage of interfaces that are relied upon by the dependent component, only that the correspondence is correct and ensuring that interfaces of the base component are mapped to interfaces required by the dependent component wherever possible. The completeness of the coverage is considered in Composition rationale (ACO_COR) activities.

18.4.3.4 Action ACO_DEV.3.2E

18.4.3.4.1 Work unit ACO_DEV.3-7

The evaluator ***shall examine*** the development information and the reliance information to determine that the interfaces are described consistently.

The evaluator's goal in this work unit is to determine that the interfaces described in the development information for the base component and the reliance information for the dependent component are represented consistently.

18.5 Reliance of dependent component (ACO_REL)

18.5.1 Evaluation of sub-activity (ACO_REL.1)

18.5.1.1 Objectives

The objectives of this sub-activity are to determine whether the developer's reliance evidence provides sufficient information to determine that the necessary functionality is available in the

base component, and the means by which that functionality is invoked. These are provided in terms of a high-level description.

18.5.1.2 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the dependent component functional specification;

c) the dependent component design;

d) the dependent component architectural design;

e) the reliance information.

18.5.1.3 Application notes

A dependent component whose TSF interacts with the base component requires functionality provided by that base component (e.g., remote authentication, remote audit data storage). In these cases, those invoked services need to be described for those charged with configuring the composed TOE for end users. The rationale for requiring this documentation is to aid integrators of the composed TOE to determine what services in the base component can have adverse effects on the dependent component, and to provide information against which to determine the compatibility of the components when applying the Development evidence (ACO_DEV) family.

18.5.1.4 Action ACO_REL.1.1E

18.5.1.4.1 General

CC Part 3 ACO_REL.1.1C: *The reliance information shall describe the functionality of the base component hardware, firmware and/or software that is relied upon by the dependent component TSF.*

18.5.1.4.2 Work unit ACO_REL.1-1

The evaluator ***shall check*** the reliance information to determine that it describes the functionality of the base dependent hardware, firmware and/or software that is relied upon by the dependent component TSF.

The evaluator assesses the description of the security functionality that the dependent component TSF requires to be provided by the base component's hardware, firmware and software. The emphasis of this work unit is on the level of detail of this description, rather than on an assessment of the information's accuracy. (The assessment of the accuracy of the information is the focus of the next work unit.)

This description of the base component's functionality need not be any more detailed than the level of the description of a component of the TSF, as would be provided in the TOE Design (TOE design (ADV_TDS))

18.5.1.4.3 Work unit ACO_REL.1-2

The evaluator ***shall examine*** the reliance information to determine that it accurately reflects the objectives specified for the operational environment of the dependent component.

The reliance information contains the description of the base component's security functionality relied upon by the dependent component. To ensure that the reliance information is consistent with the expectations of the operational environment of the dependent component, the evaluator compares the reliance information with the statement of objectives for the environment in the ST for the dependent component.

For example, if the reliance information claims that the dependent component TSF relies upon the base component to store and protect audit data, yet other evaluation evidence (e.g. the dependent component design) makes it clear that the dependent component TSF itself is storing and protecting the audit data, this would indicate an inaccuracy.

It should be noted that the objectives for the operational environment may include objectives that can be met by non-IT measures. While the services that the base component environment is expected to provide may be described in the description of IT objectives for the operational environment in the dependent component ST, it is not required that all such expectations on the environment be described in the reliance information.

CC Part 3 ACO_REL.1.2C: *The reliance information shall describe all interactions through which the dependent component TSF requests services from the base component.*

18.5.1.4.4 Work unit ACO_REL.1-3

The evaluator ***shall examine*** the reliance information to determine that it describes all interactions between the dependent component and the base component, through which the dependent component TSF requests services from the base component.

The dependent component TSF may request services of the base component that were not within the TSF of the base component (see B.3, Interactions between composed IT entities in CC Part 3).

The interfaces to the base component's functionality are described at the same level as the description of the interfaces to the dependent component TSF functionality, as would be provided between subsystems in the TOE design [Evaluation of sub-activity (ADV_TDS.1)].

The purpose of describing the interactions between the dependent component and the base component is to provide an understanding of how the dependent component TSF relies upon the base component for the provision of services to support the operation of security functionality of the dependent component. These interactions do not need to be characterised at the implementation level (e.g. parameters passed from one routine in a component to a routine in another component), but the data elements identified for a particular component that are going to be used by another component should be covered in this description. The statement should help the reader understand in general why the interaction is necessary.

Accuracy and completeness of the interfaces is based on the security functionality that the TSF requires to be provided by the base component, as assessed in work units ACO_REL.1-1 and ACO_REL.1-2. It should be possible to map all of the functionality described in the earlier work units to the interfaces identified in this work unit, and vice versa. An interface that does not correspond to described functionality would also indicate an inadequacy.

CC Part 3 ACO_REL.1.3C: *The reliance information shall describe how the dependent TSF protects itself from interference and tampering by the base component.*

18.5.1.4.5 Work unit ACO_REL.1-4

The evaluator ***shall examine*** the reliance information to determine that it describes how the dependent TSF protects itself from interference and tampering by the base component.

The description of how the dependent component protects itself from interference and tampering by the base component is to be provided at the same level of detail as necessary for ADV_ARC.1-4.

18.5.2 Evaluation of sub-activity (ACO_REL.2)

18.5.2.1 Objectives

The objectives of this sub-activity are to determine whether the developer's reliance evidence provides sufficient information to determine that the necessary functionality is available in the base component, and the means by which that functionality is invoked. This is provided in terms of the interfaces between the dependent and base component and the return values from those interfaces called by the dependent component.

18.5.2.2 Input

The evaluation evidence for this sub-activity is:

a) the composed ST;

b) the dependent component functional specification;

c) the dependent component design;

d) the dependent component implementation representation;

e) the dependent component architectural design;

f) the reliance information.

18.5.2.3 Application notes

A dependent component whose TSF interacts with the base component requires functionality provided by that base component (e.g., remote authentication, remote audit data storage). In these cases, those invoked services need to be described for those charged with configuring the composed TOE for end users. The rationale for requiring this documentation is to aid integrators of the composed TOE to determine what services in the base component can have adverse effects on the dependent component, and to provide information against which to determine the compatibility of the components when applying the Development evidence (ACO_DEV) family.

18.5.2.4 Action ACO_REL.2.1E

18.5.2.4.1 General

CC Part 3 ACO_REL.2.1C: *The reliance information shall describe the functionality of the base component hardware, firmware and/or software that is relied upon by the dependent component TSF.*

18.5.2.4.2 Work unit ACO_REL.2-1

The evaluator ***shall check*** the reliance information to determine that it describes the functionality of the base dependent hardware, firmware and/or software that is relied upon by the dependent component TSF.

The evaluator assesses the description of the security functionality that the dependent component TSF requires to be provided by the base component's hardware, firmware and software. The emphasis of this work unit is on the level of detail of this description, rather than

on an assessment of the information's accuracy. (The assessment of the accuracy of the information is the focus of the next work unit.)

This description of the base component's functionality need not be any more detailed than the level of the description of a component of the TSF, as would be provided in the TOE Design (TOE design (ADV_TDS))

18.5.2.4.3 Work unit ACO_REL.2-2

The evaluator ***shall examine*** the reliance information to determine that it accurately reflects the objectives specified for the operational environment of the dependent component.

The reliance information contains the description of the base component's security functionality relied upon by the dependent component. To ensure that the reliance information is consistent with the expectations of the operational environment of the dependent component, the evaluator compares the reliance information with the statement of objectives for the environment in the ST for the dependent component.

For example, if the reliance information claims that the dependent component TSF relies upon the base component to store and protect audit data, yet other evaluation evidence (e.g. the dependent component design) makes it clear that the dependent component TSF itself is storing and protecting the audit data, this would indicate an inaccuracy.

It should be noted that the objectives for the operational environment may include objectives that can be met by non-IT measures. While the services that the base component environment is expected to provide may be described in the description of IT objectives for the operational environment in the dependent component ST, it is not required that all such expectations on the environment be described in the reliance information.

CC Part 3 ACO_REL.2.2C: *The reliance information shall describe all interactions through which the dependent component TSF requests services from the base component.*

18.5.2.4.4 Work unit ACO_REL.2-3

The evaluator ***shall examine*** the reliance information to determine that it describes all interactions between the dependent component and the base component, through which the dependent component TSF requests services from the base component.

The dependent component TSF may request services of the base component that were not within the TSF of the base component (see CC Part 3, B.3 Interactions between composed IT entities).

The interfaces to the base component's functionality are described at the same level as the description of the interfaces to the dependent component TSF functionality, as would be provided between subsystems in the TOE design [Evaluation of sub-activity (ADV_TDS.1)].

The purpose of describing the interactions between the dependent component and the base component is to provide an understanding of how the dependent component TSF relies upon the base component for the provision of services to support the operation of security functionality of the dependent component. These interactions do not need to be characterised at the implementation level (e.g. parameters passed from one routine in a component to a routine in another component), but the data elements identified for a particular component that are going to be used by another component should be covered in this description. The statement should help the reader understand in general why the interaction is necessary.

Accuracy and completeness of the interfaces is based on the security functionality that the TSF requires to be provided by the base component, as assessed in work units ACO_REL.2-1 and ACO_REL.2-2**.** It should be possible to map all of the functionality described in the earlier work

units to the interfaces identified in this work unit, and vice versa. An interface that does not correspond to described functionality would also indicate an inadequacy.

CC Part 3 ACO_REL.2.3C: *The reliance information shall describe each interaction in terms of the interface used and the return values from those interfaces.*

18.5.2.4.5 Work unit ACO_REL.2-4

The reliance information shall describe each interaction in terms of the interface used and the return values from those interfaces.

The identification of the interfaces used by the dependent component TSF when making services requests of the base component allows an integrator to determine whether the base component provides all the necessary corresponding interfaces. This understanding is further gained through the specification of the return values expected by the dependent component. The evaluator ensures that interfaces are described for each interaction specified (as analysed in ACO_REL.2-3).

CC Part 3 ACO_REL.2.4C: *The reliance information shall describe how the dependent TSF protects itself from interference and tampering by the base component.*

18.5.2.4.6 Work unit ACO_REL.2-5

The evaluator ***shall examine*** the reliance information to determine that it describes how the dependent TSF protects itself from interference and tampering by the base component.

The description of how the dependent component protects itself from interference and tampering by the base component is to be provided at the same level of detail as necessary for ADV_ARC.1-4.

18.6 Composed TOE testing (ACO_CTT)

18.6.1 Evaluation of sub-activity (ACO_CTT.1)

18.6.1.1 Objectives

The objective of this sub-activity is to determine whether the developer correctly performed and documented tests for each of the base component interfaces on which the dependent component relies. As part of this determination the evaluator repeats a sample of the tests performed by the developer and performs any additional tests required to ensure the expected behaviour of all composed TOE SFRs and interfaces of the base component relied upon by the dependent component is demonstrated.

18.6.1.2 Input

The evaluation evidence for this sub-activity is:

a) the composed TOE suitable for testing;

b) the composed TOE testing evidence;

c) the reliance information;

d) the development information.

18.6.1.3 Action ACO_CTT.1.1E

18.6.1.3.1 General

CC Part 3 ACO_CTT.1.1C: *The composed TOE and base component interface test documentation shall consist of test plans, expected test results and actual test results.*

18.6.1.3.2 Work unit ACO_CTT.1-1

The evaluator ***shall examine*** the composed TOE test documentation to determine that it consists of test plans, expected test results and actual test results.

This work unit may be satisfied by provision of the test evidence from the evaluation of the dependent component if the base component was used to satisfy the requirements for IT in the operational environment of the dependent component.

All work units necessary for the satisfaction of ATE_FUN.1.1E will be applied to determine:

- that the test documentation consists of test plans, expected test results, and actual test results;
- that the test documentation contains the information necessary to ensure the tests are repeatable;
- the level of developer effort that was applied to testing of the base component.

18.6.1.3.3 Work unit ACO_CTT.1-2

The evaluator ***shall examine*** the base component interface test documentation to determine that it consists of test plans, expected test results and actual test results.

This work unit may be satisfied by provision of the test evidence from the evaluation of the base component for those interfaces relied upon in the composed TOE by the dependent component are TSFIs of the successfully evaluated base component. The determination of whether the interfaces of the base component relied upon by the dependent component were in fact TSFIs of the evaluated base component is made during the ACO_COR activity.

All work units necessary for the satisfaction of ATE_FUN.1.1E will be applied to determine:

a) that the test documentation consist of test plans expected test results and actual test results;

b) that the test documentation contains the information necessary to ensure the tests are repeatable;

c) the level of developer effort that was applied to testing of the base component.

CC Part 3 ACO_CTT.1.2C: *The test documentation from the developer execution of the composed TOE tests shall demonstrate that the TSF behaves as specified.*

18.6.1.3.4 Work unit ACO_CTT.1-3

The evaluator ***shall examine*** the test documentation to determine that the developer execution of the composed TOE tests shall demonstrate that the TSF behaves as specified.

The evaluator should construct a mapping between the tests described in the test plan and the SFRs specified for the composed TOE to identify which SFRs have been tested by the developer.

Guidance on this work unit can be found in:

- Subclause 15.2.1.

- Subclause 15.2.2.

The outputs from the successful execution of the tests (as assessed for ATE_FUN.1.3C can be compared with the mapping to determine that the SFRs of the composed TOE, as tested by the developer, behave as expected.

CC Part 3 ACO_CTT.1.3C: *The test documentation from the developer execution of the base component interface tests shall demonstrate that the base component interface relied upon by the dependent component behaves as specified.*

18.6.1.3.5 Work unit ACO_CTT.1-4

The evaluator ***shall examine*** the test documentation to determine that the developer execution of the base component interface tests shall demonstrate that the base component interfaces relied upon by the dependent component behave as specified.

The evaluator should construct a mapping between the tests described in the test plan and the interfaces of the base component relied upon by the dependent component (as specified in the reliance information, examined under ACO_REL) to identify which base component interfaces have been tested by the developer.

Guidance on this work unit can be found in:

- Subclause 15.2.1.
- Subclause 15.2.2.

The outputs from the successful execution of the tests (as assessed for ATE_FUN.1.3C can be compared with the mapping to determine that the interfaces of the base component, as tested by the developer, behave as expected.

CC Part 3 ACO_CTT.1.4C: *The base component shall be suitable for testing.*

18.6.1.3.6 Work unit ACO_CTT.1-5

The evaluator ***shall examine*** the composed TOE to determine that it has been installed properly and is in a known state.

To determine that the composed TOE has been installed properly and is in a known state the ATE_IND.2-1 and ATE_IND.2-2 work units will be applied to the TOE provided by the developer for testing.

18.6.1.3.7 Work unit ACO_CTT.1-6

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 base component developer to functionally test the base component.

To determine that the set of resources provided are equivalent to those used to functionally test the base component as used in the composed TOE, the ATE_IND.2-3 work unit will be applied.

18.6.1.4 Action ACO_CTT.1.2E

18.6.1.4.1 Work unit ACO_CTT.1-7

The evaluator ***shall perform*** testing in accordance with ATE_IND.2.2E, for a subset of the SFRs specified in the composed security target, to verify the developer test results.

The evaluator will apply all work units necessary for the satisfaction of ATE_IND.2.2E, reporting in the ETR for the composed TOE all analysis, results and verdicts as dictated by the associated work units.

18.6.1.5 Action ACO_CTT.1.3E

18.6.1.5.1 Work unit ACO_CTT.1-8

The evaluator ***shall perform*** testing in accordance with ATE_IND.2.3E, for a subset of the SFRs specified in the composed security target, to confirm that the TSF operates as specified.

The evaluator will apply all work units necessary for the satisfaction of ATE_IND.2.3E, reporting in the ETR for the composed TOE all analysis, results and verdicts as dictated by the work units.

When selecting interfaces of the TSF of the composed TOE to test, the evaluator should take into account any modifications to the components from the evaluated version or configuration. Modifications to the component from that evaluated may include patches introduced, a different configuration as a result of modified guidance documentation, reliance an additional portion of the component that was not within the TSF of the component. These modifications will have been identified during the Composition rationale (ACO_COR) activity.

18.6.2 Evaluation of sub-activity (ACO_CTT.2)

18.6.2.1 Objectives

The objective of this sub-activity is to determine whether the developer correctly performed and documented tests for each of the base component interfaces on which the dependent component relies. As part of this determination the evaluator repeats a sample of the tests performed by the developer and performs any additional tests required to fully demonstrate the expected behaviour of the composed TOE and the interfaces of the base component relied upon by the dependent component.

18.6.2.2 Input

The evaluation evidence for this sub-activity is:

a) the composed TOE suitable for testing;

b) the composed TOE testing evidence;

c) the reliance information;

d) the development information.

18.6.2.3 Action ACO_CTT.2.1E

18.6.2.3.1 General

CC Part 3 ACO_CTT.2.1C: *The composed TOE and base component interface test documentation shall consist of test plans, expected test results and actual test results.*

18.6.2.3.2 Work unit ACO_CTT.2-1

The evaluator ***shall examine*** the composed TOE test documentation to determine that it consists of test plans, expected test results and actual test results.

This work unit may be satisfied by provision of the test evidence from the evaluation of the dependent component if the base component was used to satisfy the requirements for IT in the operational environment of the dependent component.

All work units necessary for the satisfaction of ATE_FUN.1.1E will be applied to determine:

a) that the test documentation consist of test plans expected test results and actual test results;

b) that the test documentation contains the information necessary to ensure the tests are repeatable;

c) the level of developer effort that was applied to testing of the base component.

18.6.2.3.3 Work unit ACO_CTT.2-2

The evaluator ***shall examine*** the base component interface test documentation to determine that it consists of test plans, expected test results and actual test results.

This work unit may be satisfied by provision of the test evidence from the evaluation of the base component for those interfaces relied upon in the composed TOE by the dependent component are TSFIs of the successfully evaluated base component. The determination of whether the interfaces of the base component relied upon by the dependent component were in fact TSFIs of the evaluated base component is made during the ACO_COR activity.

All work units necessary for the satisfaction of ATE_FUN.1.1E will be applied to determine:

a) that the test documentation consists of test plans, expected test results, and actual test results;

b) that the test documentation contains the information necessary to ensure the tests are repeatable;

c) the level of developer effort that was applied to testing of the base component.

CC Part 3 ACO_CTT.2.2C: *The test documentation from the developer execution of the composed TOE tests shall demonstrate that the TSF behaves as specified and is complete.*

18.6.2.3.4 Work unit ACO_CTT.2-3

The evaluator ***shall examine*** the test documentation to determine that it provides accurate correspondence between the tests in the test documentation relating to the testing of the composed TOE and the composed TOE SFRs in the composed TOE security target.

A simple cross-table may be sufficient to show test correspondence. The identification of correspondence between the tests and SFRs presented in the test documentation has to be unambiguous.

18.6.2.3.5 Work unit ACO_CTT.2-4

The evaluator ***shall examine*** the test documentation to determine that the developer execution of the composed TOE tests shall demonstrate that the TSF behaves as specified.

Guidance on this work unit can be found in:

- Subclause 15.2.1.
- Subclause 15.2.2.

The outputs from the successful execution of the tests (as assessed for ATE_FUN.1.3C can be compared with the mapping to determine that the SFRs of the composed TOE, as tested by the developer, behave as expected.

CC Part 3 ACO_CTT.2.3C: *The test documentation from the developer execution of the base component interface tests shall demonstrate that the base component interface relied upon by the dependent component behaves as specified and is complete.*

18.6.2.3.6 Work unit ACO_CTT.2-5

The evaluator ***shall examine*** the test documentation to determine that it provides accurate correspondence between the tests in the test documentation relating to the testing of the base component interfaces relied upon by the dependent component and the interfaces specified in the reliance information.

A simple cross-table may be sufficient to show test correspondence. The identification of correspondence between the tests and interfaces presented in the test documentation has to be unambiguous.

18.6.2.3.7 Work unit ACO_CTT.2-6

The evaluator ***shall examine*** the test documentation to determine that the developer execution of the base component interface tests shall demonstrate that the base component interfaces relied upon by the dependent component behave as specified.

Guidance on this work unit can be found in:

- Subclause 15.2.1**.**
- Subclause 15.2.2.

The outputs from the successful execution of the tests (as assessed for ATE_FUN.1.3C can be compared with the mapping to determine that the interfaces of the base component, as tested by the developer, behave as expected.

CC Part 3 ACO_CTT.2.4C: *The base component shall be suitable for testing.*

18.6.2.3.8 Work unit ACO_CTT.2-7

The evaluator ***shall examine*** the composed TOE to determine that it has been installed properly and is in a known state.

To determine that the composed TOE has been installed properly and is in a known state the ATE_IND.2-1 and ATE_IND.2-2 work units will be applied to the TOE provided by the developer for testing.

18.6.2.3.9 Work unit ACO_CTT.2-8

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 base component developer to functionally test the base component.

To determine that the set of resources provided are equivalent to those used to functionally test the base component as used in the composed TOE, the ATE_IND.2-3 work unit will be applied.

18.6.2.4 Action ACO_CTT.2.2E

18.6.2.4.1 Work unit ACO_CTT.2-9

The tests are to be selected and executed in accordance with ATE_IND.2.2E, to demonstrate the correct behaviour of the SFRs specified in the composed TOE security target.

The evaluator will apply all work units necessary for the satisfaction of ATE_IND.2.2E, reporting in the ETR for the composed TOE all analysis, results and verdicts as dictated by the associated work units.

18.6.2.5 Action ACO_CTT.2.3E

18.6.2.5.1 Work unit ACO_CTT.2-10

The evaluator ***shall perform*** testing in accordance with ATE_IND.2.3E, for a subset of the SFRs specified in the composed security target, to confirm that the TSF operates as specified.

The evaluator will apply all work units necessary for the satisfaction **of** ATE_IND.2.3E, reporting in the ETR for the composed TOE all analysis, results and verdicts as dictated by the work units.

When selecting interfaces of the TSF of the composed TOE to test, the evaluator should take into account any modifications to the components from the evaluated version or configuration. Modifications to the component from that evaluated may include patches introduced, a different configuration as a result of modified guidance documentation, reliance an additional portion of the component that was not within the TSF of the component. These modifications will have been identified during the Composition rationale (ACO_COR) activity.

18.6.2.5.2 Work unit ACO_CTT.2-11

The evaluator ***shall perform*** testing, in accordance with Evaluation of sub-activity (ATE_IND.2), for a subset of the interfaces to the base component to confirm they operate as specified.

The evaluator will apply all work units necessary for the satisfaction of ATE_IND.2.3E, reporting in the ETR for the composed TOE all analysis, results and verdicts as dictated by the work units.

When selecting interfaces of the base component to test, the evaluator should take into account any modifications to the base component from the evaluated version or configuration. In particular, the evaluator should consider the development of tests to demonstrate the correct behaviour of interfaces of the base component that were not considered during the evaluation of the base component. These additional interfaces and other modifications to the base component will have been identified during the Composition rationale (ACO_COR) activity.

18.7 Composition vulnerability analysis (ACO_VUL)

18.7.1 Evaluation of sub-activity (ACO_VUL.1)

18.7.1.1 Objectives

The objective of this sub-activity is to determine whether the composed TOE, in its operational environment, has easily exploitable vulnerabilities.

The developer provides details of any residual vulnerabilities reported from evaluation of the components. The evaluator performs an analysis of the disposition the residual vulnerabilities reported and also performs a search of the public domain, to identify any new potential vulnerabilities in the components (i.e. those issues that have been reported in the public domain since evaluation of the base component). The evaluator then performs penetration testing to demonstrate that the potential vulnerabilities cannot be exploited in the TOE, in its operational environment, by an attacker with basic attack potential.

18.7.1.2 Input

The evaluation evidence for this sub-activity is:

a) the composed TOE suitable for testing;

b) the composed ST;

c) the composition rationale;

d) the guidance documentation;

e) information publicly available to support the identification of possible security vulnerabilities;

f) residual vulnerabilities reported during evaluation of each component.

18.7.2 Application notes

See the application notes for Evaluation of sub-activity (AVA_VAN.1).

18.7.2.1 Action ACO_VUL.1.1E

18.7.2.1.1 General

CC Part 3 ACO_VUL.1.1C: *The composed TOE shall be suitable for testing.*

18.7.2.1.2 Work unit ACO_VUL.1-1

The evaluator ***shall examine*** the composed TOE to determine that it has been installed properly and is in a known state.

To determine that the composed TOE has been installed properly and is in a known state the ATE_IND.2-1 and ATE_IND.2-2 work units will be applied to the composed TOE.

If the assurance package includes a component from the ACO_CTT family, then the evaluator may refer to the result of the work unit ACO_CTT*-1 to demonstrate this has been satisfied.

18.7.2.1.3 Work unit ACO_VUL.1-2

The evaluator ***shall examine*** the composed TOE configuration to determine that any assumptions and objectives in the STs the components relating to IT entities for are fulfilled by the other components.

The STs for the component may include assumptions about other components that may use the component to which the ST relates, e.g. the ST for an operating system used as a base component may include an assumption that any applications loaded on the operating system do not run in privileged mode. These assumptions and objectives are to be fulfilled by other components in the composed TOE.

18.7.2.2 Action ACO_VUL.1.2E

18.7.2.2.1 Work unit ACO_VUL.1-3

The evaluator ***shall examine*** the residual vulnerabilities from the base component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the base component, which were demonstrated to be non-exploitable in the base component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the base component it was assumed that a particular operating system service was disabled, which is enabled in the composed TOE evaluation, any potential vulnerabilities relating to that service previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the base component should be considered in the light of any known, non-exploitable vulnerabilities for the other components (e.g. dependent component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.2.2.2 Work unit ACO_VUL.1-4

The evaluator ***shall examine*** the residual vulnerabilities from the dependent component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the dependent component, which were demonstrated to be non-exploitable in the dependent component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the dependent component it was assumed that IT meeting the operational environment requirements would not return a certain value in response to a service request, which is provided by the base component in the composed TOE evaluation, any potential vulnerabilities relating to that return value previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the dependent component should be considered in the light of any known, non-exploitable vulnerabilities for the other components (e.g. base component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.2.3 Action ACO_VUL.1.3E

18.7.2.3.1 Work unit ACO_VUL.1-5

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the base component that have become known since the completion of evaluation of the base component.

The evaluator will use the information in the public domain as described in AVA_VAN.1-2 to search for vulnerabilities in the base component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the base component do not have to be further investigated unless it is apparent to the evaluator that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since the base component evaluation that means the exploitation of the potential vulnerability has been simplified.

18.7.2.3.2 Work unit ACO_VUL.1-6

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the dependent component that have become known since the completion of the dependent component evaluation.

The evaluator will use the information in the public domain as described in AVA_VAN.1-2 to search for vulnerabilities in the dependent component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the dependent component do not have to be further investigated unless it is apparent to the evaluator

that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since evaluation of the dependent component that means the exploitation of the potential vulnerability has been simplified.

18.7.2.3.3 Work unit ACO_VUL.1-7

The evaluator ***shall record*** in the ETR the identified potential security vulnerabilities that are candidates for testing and applicable to the composed TOE in its operational environment.

The ST, guidance documentation and functional specification are used to determine whether the vulnerabilities are relevant to the composed TOE in its operational environment.

The evaluator records any reasons for exclusion of vulnerabilities from further consideration if the evaluator determines that the vulnerability is not applicable in the operational environment. Otherwise the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the composed TOE in its operational environment, which can be used as an input into penetration testing activities (i.e. ACO_VUL.1.4E), shall be reported in the ETR by the evaluators.

18.7.2.4 Action ACO_VUL.1.4E

18.7.2.4.1 Work unit ACO_VUL.1-8

The evaluator ***shall conduct*** penetration testing as detailed for AVA_VAN.1.3E.

The evaluator will apply all work units necessary for the satisfaction of evaluator action AVA_VAN.1.3E, reporting in the ETR for the composed TOE all analysis and verdicts as dictated by the work units.

The evaluator will also apply the work units for the evaluator **action** AVA_VAN.1.1E to determine that the composed TOE provided by the developer is suitable for testing.

18.7.3 Evaluation of sub-activity (ACO_VUL.2)

18.7.3.1 Objectives

The objective of this sub-activity is to determine whether the composed TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing basic attack potential.

The developer provides an analysis of the disposition of any residual vulnerabilities reported for the components and of any vulnerabilities introduced through the combination of the base and dependent components. The evaluator performs a search of the public domain to identify any new potential vulnerabilities in the components (i.e. those issues that have been reported in the public domain since the completion of the evaluation of the components). The evaluator will also perform an independent vulnerability analysis of the composed TOE and penetration testing.

18.7.3.2 Input

The evaluation evidence for this sub-activity is:

a) the composed TOE suitable for testing;

b) the composed ST;

c) the composition rationale;

d) the reliance information;

e) the guidance documentation;

f) information publicly available to support the identification of possible security vulnerabilities;

g) residual vulnerabilities reported during evaluation of each component.

18.7.3.3 Application notes

See the application notes for Evaluation of sub-activity (AVA_VAN.2).

18.7.3.4 Action ACO_VUL.2.1E

18.7.3.4.1 General

CC Part 3 ACO_VUL.2.1C: *The composed TOE shall be suitable for testing.*

18.7.3.4.2 Work unit ACO_VUL.2-1

The evaluator ***shall examine*** the composed TOE to determine that it has been installed properly and is in a known state.

To determine that the composed TOE has been installed properly and is in a known state the ATE_IND.2-1 and ATE_IND.2-2 work units will be applied to the composed TOE.

If the assurance package includes ACO_CTT family, then the evaluator may refer to the result of the work unit Composed TOE testing (ACO_CTT)*-1 to demonstrate this has been satisfied.

18.7.3.4.3 Work unit ACO_VUL.2-2

The evaluator ***shall examine*** the composed TOE configuration to determine that any assumptions and objectives in the STs the components relating to IT entities for are fulfilled by the other components.

The STs for the component may include assumptions about other components that may use the component to which the ST relates, e.g. the ST for an operating system used as a base component may include an assumption that any applications loaded on the operating system do not run in privileged mode. These assumptions and objectives are to be fulfilled by other components in the composed TOE.

18.7.3.5 Action ACO_VUL.2.2E

18.7.3.5.1 Work unit ACO_VUL.2-3

The evaluator ***shall examine*** the residual vulnerabilities from the base component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the base component, which were demonstrated to be non-exploitable in the base component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the base component it was assumed that a particular operating system service was disabled, which is enabled in the composed TOE evaluation, any potential vulnerabilities relating to that service previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the base component should be considered in the light of any known, non-exploitable vulnerabilities for the

other components (e.g. dependent component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.3.5.2 Work unit ACO_VUL.2-4

The evaluator ***shall examine*** the residual vulnerabilities from the dependent component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the dependent component, which were demonstrated to be non-exploitable in the dependent component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the dependent component it was assumed that IT meeting the operational environment requirements would not return a certain value in response to a service request, which is provided by the base component in the composed TOE evaluation, any potential vulnerabilities relating to that return value previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the dependent component should be considered in the light of any known, non-exploitable vulnerabilities for the other components (e.g. base component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.3.6 Action ACO_VUL.2.3E

18.7.3.6.1 Work unit ACO_VUL.2-5

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the base component that have become known since the completion of the base component evaluation.

The evaluator will use the information in the public domain as described in AVA_VAN.2-2 to search for vulnerabilities in the base component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the base component do not have to be further investigated unless it is apparent to the evaluator that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since the base component evaluation that means the exploitation of the potential vulnerability has been simplified.

18.7.3.6.2 Work unit ACO_VUL.2-6

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the dependent component that have become known since the completion of the dependent component evaluation.

The evaluator will use the information in the public domain as described in AVA_VAN.2-2 to search for vulnerabilities in the dependent component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the dependent component do not have to be further investigated unless it is apparent to the evaluator that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since

evaluation of the dependent component that means the exploitation of the potential vulnerability has been simplified.

18.7.3.6.3 Work unit ACO_VUL.2-7

The evaluator ***shall record*** in the ETR the identified potential security vulnerabilities that are candidates for testing and applicable to the composed TOE in its operational environment.

The ST, guidance documentation and functional specification are used to determine whether the vulnerabilities are relevant to the composed TOE in its operational environment.

The evaluator records any reasons for exclusion of vulnerabilities from further consideration if the evaluator determines that the vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the composed TOE in its operational environment, which can be used as an input into penetration testing activities (ACO_VUL.2.5E), shall be reported in the ETR by the evaluators.

18.7.3.7 Action ACO_VUL.2.4E

18.7.3.7.1 Work unit ACO_VUL.2-8

The evaluator ***shall conduct*** a search of the composed TOE ST, guidance documentation, reliance information and composition rationale to identify possible security vulnerabilities in the composed TOE.

The consideration of the components of the composed TOE in the independent evaluator vulnerability analysis will take a slightly different form to that documented in AVA_VAN.2.3E for a component evaluation, as it will not necessarily consider all layers of design abstraction relevant to the assurance package. These will have already been considered during the evaluation of the components, but the evidence may not be available for the composed TOE evaluation. However, the general approach described in the work units associated with AVA_VAN.2.3E is applicable and should form the basis of the evaluator's search for potential vulnerabilities in the composed TOE.

A vulnerability analysis of the individual components used in the composed TOE will have already been performed during evaluation of the individual components. The focus of the vulnerability analysis during the composed TOE evaluation is to identify any vulnerabilities introduced as a result of the integration of the components or due to any changes in the use of the components between the evaluated component configuration to the composed TOE configuration.

The evaluator will use the understanding of the component's construction as detailed in the reliance information for the dependent component, and the development information and composition rationale for the base component, together with the dependent component design information. This information will allow the evaluator to gain an understanding of how the base component and dependent component interact and identify potential vulnerabilities that may be introduced as a result of this interaction.

The evaluator will consider any new guidance provided for the installation, start-up and operation of the composed TOE to identify any potential vulnerabilities introduced through this revised guidance.

If any of the individual components have been through assurance continuity activities since the completion of the component evaluation, the evaluator will consider the patch(es) in the independent vulnerability analysis. Information related to the change provided in a public report of the assurance continuity activities (e.g. Maintenance Report) will be the main source of input

material of the change. This will be supplemented by any updates to the guidance documentation resulting from the change and any information regarding the change available in the public domain, e.g. vendor website.

Any risks identified due to the lack of evidence to establish the full impact of any patches or deviations in the configuration of a component from the evaluated configuration are to be documented in the evaluator's vulnerability analysis.

18.7.3.8 Action ACO_VUL.2.5E

18.7.3.8.1 Work unit ACO_VUL.2-9

The evaluator ***shall conduct*** penetration testing as detailed for AVA_VAN.2.4E.

The evaluator will apply all work units necessary for the satisfaction of evaluator action AVA_VAN.2.4E, reporting in the ETR for the composed TOE all analysis and verdicts as dictated by the work units.

The evaluator will also apply the work units for the evaluator action AVA_VAN.2.1E to determine that the composed TOE provided by the developer is suitable for testing.

18.7.4 Evaluation of sub-activity (ACO_VUL.3)

18.7.4.1 Objectives

The objective of this sub-activity is to determine whether the composed TOE, in its operational environment, has vulnerabilities exploitable by attackers possessing Enhanced-Basic attack potential.

The developer provides an analysis of the disposition of any residual vulnerabilities reported for the components and of any vulnerabilities introduced through the combination of the base and dependent components. The evaluator performs a search of the public domain to identify any new potential vulnerabilities in the components (i.e. those issues that have been reported in the public domain since the completion of the component evaluations). The evaluator will also perform an independent vulnerability analysis of the composed TOE and penetration testing.

18.7.4.2 Input

The evaluation evidence for this sub-activity is:

a) the composed TOE suitable for testing;

b) the composed ST;

c) the composition rationale;

d) the reliance information;

e) the guidance documentation;

f) information publicly available to support the identification of possible security vulnerabilities;

g) residual vulnerabilities reported during evaluation of each component.

18.7.4.3 Application notes

See the application notes for Evaluation of sub-activity (AVA_VAN.3).

18.7.4.4 Action ACO_VUL.3.1E

18.7.4.4.1 General

CC Part 3 ACO_VUL.3.1C: *The composed TOE shall be suitable for testing.*

18.7.4.4.2 Work unit ACO_VUL.3-1

The evaluator ***shall examine*** the composed TOE to determine that it has been installed properly and is in a known state.

To determine that the composed TOE has been installed properly and is in a known state the ATE_IND.2-1 and ATE_IND.2-2 work units will be applied to the composed TOE.

If the assurance package includes ACO_CTT family, then the evaluator may refer to the result of the work unit Composed TOE testing (ACO_CTT)*-1 to demonstrate this has been satisfied.

18.7.4.4.3 Work unit ACO_VUL.3-2

The evaluator ***shall examine*** the composed TOE configuration to determine that any assumptions and objectives in the STs the components relating to IT entities for are fulfilled by the other components.

The STs for the component may include assumptions about other components that may use the component to which the ST relates, e.g. the ST for an operating system used as a base component may include an assumption that any applications loaded on the operating system do not run in privileged mode. These assumptions and objectives are to be fulfilled by other components in the composed TOE.

18.7.4.5 Action ACO_VUL.3.2E

18.7.4.5.1 Work unit ACO_VUL.3-3

The evaluator ***shall examine*** the residual vulnerabilities from the base component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the base component, which were demonstrated to be non-exploitable in the base component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the base component it was assumed that a particular operating system service was disabled, which is enabled in the composed TOE evaluation, any potential vulnerabilities relating to that service previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the base component should be considered in the light of any known, non-exploitable vulnerabilities for the other components (e.g. dependent component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.4.5.2 Work unit ACO_VUL.3-4

The evaluator ***shall examine*** the residual vulnerabilities from the dependent component evaluation to determine that they are not exploitable in the composed TOE in its operational environment.

The list of vulnerabilities identified in the product during the evaluation of the dependent component, which were demonstrated to be non-exploitable in the dependent component, is to be used as an input into this activity. The evaluator will determine that the premise(s) on which a vulnerability was deemed to be non-exploitable is upheld in the composed TOE, or whether the combination has re-introduced the potential vulnerability. For example, if during evaluation of the dependent component it was assumed that IT meeting the operational environment requirements would not return a certain value in response to a service request, which is provided by the base component in the composed TOE evaluation, any potential vulnerabilities relating to that return value previously scoped out should now be considered.

Also, this list of known, non-exploitable vulnerabilities resulting from the evaluation of the dependent component should be considered in the light of any known, non-exploitable vulnerabilities for the other components (e.g. base component) within the composed TOE. This is to consider the case where a potential vulnerability that is non-exploitable in isolation is exploitable when integrated with an IT entity containing another potential vulnerability.

18.7.4.6 Action ACO_VUL.3.3E

18.7.4.6.1 Work unit ACO_VUL.3-5

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the base component that have become known since the completion of the base component evaluation.

The evaluator will use the information in the public domain as described in AVA_VAN.3-2 to search for vulnerabilities in the base component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the base component do not have to be further investigated unless it is apparent to the evaluator that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since the base component evaluation that means the exploitation of the potential vulnerability has been simplified.

18.7.4.6.2 Work unit ACO_VUL.3-6

The evaluator ***shall examine*** the sources of information publicly available to support the identification of possible security vulnerabilities in the dependent component that have become known since completion of the dependent component evaluation.

The evaluator will use the information in the public domain as described in AVA_VAN.3-2 to search for vulnerabilities in the dependent component.

Those potential vulnerabilities that were publicly available prior to the evaluation of the dependent component do not have to be further investigated unless it is apparent to the evaluator that the attack potential required by an attacker to exploit the potential vulnerability has been significantly reduced. This may be through the introduction of some new technology since evaluation of the dependent component that means the exploitation of the potential vulnerability has been simplified.

18.7.4.6.3 Work unit ACO_VUL.3-7

The evaluator ***shall record*** in the ETR the identified potential security vulnerabilities that are candidates for testing and applicable to the composed TOE in its operational environment.

The ST, guidance documentation and functional specification are used to determine whether the vulnerabilities are relevant to the composed TOE in its operational environment.

The evaluator records any reasons for exclusion of vulnerabilities from further consideration if the evaluator determines that the vulnerability is not applicable in the operational environment. Otherwise, the evaluator records the potential vulnerability for further consideration.

A list of potential vulnerabilities applicable to the composed TOE in its operational environment, which can be used as an input into penetration testing activities (ACO_VUL.3.5E), shall be reported in the ETR by the evaluators.

18.7.4.7 Action ACO_VUL.3.4E

18.7.4.7.1 Work unit ACO_VUL.3-8

The evaluator ***shall conduct*** a search of the composed TOE ST, guidance documentation, reliance information and composition rationale to identify possible security vulnerabilities in the composed TOE.

The consideration of the components in the independent evaluator vulnerability analysis will take a slightly different form to that documented in AVA_VAN.3.3E for a component evaluation, as it will not necessarily consider all layers of design abstraction relevant to the assurance package. These will have already been considered during the evaluation of the base component, but the evidence may not be available for the composed TOE evaluation. However, the general approach described in the work units associated with AVA_VAN.3.3E is applicable and should form the basis of the evaluator's search for potential vulnerabilities in the composed TOE.

A vulnerability analysis of the individual components used in the composed TOE will have already been performed during evaluation of the components. The focus of the vulnerability analysis during the composed TOE evaluation is to identify any vulnerabilities introduced as a result of the integration of the components or due to any changes in the use of the components between the configuration of the component determined during the component evaluation and the composed TOE configuration.

The evaluator will use the understanding of the component's construction as detailed in the reliance information for the dependent component, and the composition rationale and development information for the base component, together with the dependent component design information. This information will allow the evaluator to gain an understanding of how the base component and dependent component interact.

The evaluator will consider any new guidance provided for the installation, start-up and operation of the composed TOE to identify any potential vulnerabilities introduced through this revised guidance.

If any of the individual components have been through assurance continuity activities since the completion of the component evaluation, the evaluator will consider the patch in the independent vulnerability analysis. Information related to the change provided in a public report of the assurance continuity activities (e.g. Maintenance Report). This will be supplemented by any updates to the guidance documentation resulting from the change and any information regarding the change available in the public domain, e.g. vendor website.

Any risks identified due to the lack of evidence to establish the full impact of any patches or deviations in the configuration of a component from the evaluated configuration are to be documented in the evaluator's vulnerability analysis.

18.7.4.8 Action ACO_VUL.3.5E

18.7.4.8.1 Work unit ACO_VUL.3-9

The evaluator ***shall conduct*** penetration testing as detailed for AVA_VAN.3.4E.

The evaluator will apply all work units necessary for the satisfaction of evaluator action AVA_VAN.3.4E, reporting in the ETR for the composed TOE all analysis and verdicts as dictated by the work units.

The evaluator will also apply the work units for the evaluator action AVA_VAN.3.1E to determine that the composed TOE provided by the developer is suitable for testing.

Annex A
(informative)

General evaluation guidance

A.1 Objectives

The objective of this clause 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.

A.2 Sampling

This Subclause 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.

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:

The subset reveals no errors, allowing the evaluator to have some confidence that the entire set is correct.

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.

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.

Sampling in the cases identified in the CC, and in cases specifically covered in evaluation methodology 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.

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 can 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.

General evaluation guidance

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, then 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 for sampling a developer's test results will differ. 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.

In certain cases it may be appropriate for the evaluator to give greater emphasis to the repetition of developer testing. For example, if the independent tests left for the evaluator to perform would be only superficially different from those included in an extensive developer test set (possibly because the developer has performed more testing than necessary to satisfy the Coverage (ATE_COV) and Depth (ATE_DPT) criteria) then it would be appropriate for the evaluator to give greater focus to the repetition of developer tests. Note that this does not necessarily imply a requirement for a high percentage sample for repetition of developer tests; indeed, given an extensive developer test set, the evaluator may be able to justify a low percentage sample.

Where the developer has used an automated test suite to perform functional testing, it will usually be easier for the evaluator to re-run the entire test suite rather than repeat only a sample of developer tests. However, the evaluator does have an obligation to check that the automatic testing does not give misrepresentative results. The implication is thus that this check must be performed for a sample of the automatic test suite, with the principles for selecting some tests in preference to others and ensuring a sufficient sample size applying equally in this case.

The principles in the list below should be followed whenever sampling is performed:

a) Sampling should not be random, rather it should be chosen such that it is representative of all of the evidence. The sample size and composition must always be justified;

b) When sampling relates to the correct implementation of the TOE, the sample should be representative of all aspects relevant to the areas that are sampled. In particular, the selection should cover a variety of components, interfaces, developer and operational sites (if more than one is involved) and hardware platform types (if more than one is involved). 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);

c) Also, when sampling relates to specifically gaining evidence that the developer testing is repeatable and reproducible the sample used must be sufficient to represent all distinct aspects of developer testing, such as different test regimes. The sample used must be sufficient to detect any systematic problem in the developer's functional testing process. The evaluator contribution resulting from the combination of repeating developer tests and performing independent tests must be sufficient to address the major points of concern for the TOE;

d) Where sampling relates to gaining evidence that a process (e.g. visitor control or design review) the evaluator should sample sufficient information to gain reasonable confidence that the procedure is being followed;

e) 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;

f) 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.

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 can 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.

A.3 Dependencies

A.3.1 General

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 Subclause provides general guidance on dependencies between different activities, sub-activities, and actions.

A.3.2 Dependencies between activities

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.

A.3.3 Dependencies between sub-activities

Dependencies identified between components in CC Part 3 have to be considered by the evaluator. Most dependencies are one way, e.g. Evaluation of sub-activity (AVA_VAN.1) claims a dependency on Evaluation of sub-activity (ADV_FSP.1) and Evaluation of sub-activity (AGD_OPE.1). There are also instances of mutual dependencies, where both components depend on each other. An example of this is Evaluation of sub-activity (ATE_FUN.1) and Evaluation of sub-activity (ATE_COV.1).

A sub-activity can be assigned a pass verdict normally only if all those sub-activities are successfully completed on which it has a one-way dependency. For example, a pass verdict on Evaluation of sub-activity (AVA_VAN.1) can normally only be assigned if the sub-activities related to Evaluation of sub-activity (ADV_FSP.1) and Evaluation of sub-activity (AGD_OPE.1) are assigned a pass verdict too. In the case of mutual dependency the ordering of these components is down to the evaluator deciding which sub-activity to perform first. Note that this indicates that pass verdicts can normally only be assigned once both sub-activities have been successful.

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.

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.

A.3.4 Dependencies between actions

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.

A.4 Site Visits

A.4.1 General

The assurance class ALC includes requirements for:

a) the application of configuration management, ensuring that the integrity of the TOE is preserved;

b) measures, procedures, and standards concerned with secure delivery of the TOE, ensuring that the security protection offered by the TOE is not compromised during the transfer to the user;

c) security measures, used to protect the development environment.

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.

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 as described in the delivery documentation;

c) to observe the application of security measures during development and maintenance of the TOE as described in the development security documentation.

Specific and detailed information is given in work units for those activities where site visits are performed:

a) CM capabilities (ALC_CMC).n with n>=3 (especially work unit ALC_CMC.3-10 = ALC_CMC.4-13 = ALC_CMC.5-19);

b) Delivery (ALC_DEL) (especially work unit ALC_DEL.1-2);

c) Development security (ALC_DVS) (especially work unit ALC_DVS.1-3 = ALC_DVS.2-4).

A.4.2 General approach

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 can 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 can occur at multiple facilities within a single building, multiple buildings at the same site, or at multiple sites.

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 can allow corrective measures to be put in place if serious deficiencies in the applied procedures emerge. This avoids unnecessary evaluation effort.

Interviews are also a useful means of determining whether the written procedures reflect what is done. In conducting such interviews, the evaluator aims to gain a deeper understanding of the analysed procedures at the development site, how they are used in practise 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.

As a first step preparing the site visits the evaluators should perform the evaluator work units concerning the assurance class ALC excluding the aspects describing the results of the site visit. Based on the information provided by the relevant developer documentation and the remaining open questions which were not answered by the documentation the evaluators compile a check list of the questions which are to be resolved by the site visits.

The first version of the evaluation report concerning the ALC class and the check list serves as input for the consultation with the evaluation authority concerning the site visits.

The check list serves as a guideline for the site visits, which questions are to be answered by inspection of the relevant measures, their application and results, and by interviews. Where appropriate, sampling is used for gaining the required level of confidence (see A.2).

The results of the site visits are recorded and serve as input for the final version of the evaluation report concerning the assurance class ALC.

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 evaluation authority. Appropriate security criteria and a methodology should be based on other standards of the Information Security Management Systems area.

A.5 Orientation guide for the preparation of the checklist

In the following some keywords are provided regarding which topics should be checked during an audit.

A.5.1 Aspects of configuration management

Basic:

- items of the configuration list, including TOE, source code, run time libraries, design documentation, development tools (ALC_CMC.3-8);
- tracking of design documentation, source code, user guidance to different versions of the TOE;
- integration of the configuration system in the design and development process, test planning, test analysis and quality management procedures.

Test analysis:

- tracking of test plans and results to specific configurations and versions of the TOE;
- access control to development systems;

- policies for access control and logging;
- policies for project specific assignment and changing of access rights.

Clearance:

- policies for clearance of the TOE and user guidance to the customer;
- policies for testing and approving of components and the TOE before deployment.

A.5.2 Aspects of development security

- infrastructure

Security measures for physical access control to the development site and rationale for the effectiveness of these measures.

Organisational measures:

- organisational structure of the company in respect of the security of the development environment;
- organisational separation between development, production, testing and quality assurance.

Personal measures:

- measures for education of the personnel in respect of development security;
- measures and legal agreements of non-disclosure of internal information.

Access control:

- assignment of secured objects (for instance TOE, source code, run time libraries, design documentation, development tools, user guidance) and security policies;
- policies and responsibilities concerning the access control and the handling of authentication information;
- policies for logging of any kind access to the development site and protection of the logging data.

Input, processing and output of data:

- security measures for protection of output and output devices (printer, plotter and displays);
- securing of local networks and communication connections.

Storage, transfer and destruction of documents and data media:

- policies for handling of documents and data media;
- policies and responsibilities for destruction of sorted out documents and logging of these events.

Data protection:

- policies and responsibilities for data and information protection (e.g. for performing backups).

Contingency plan:

- practices in case of emergency and responsibilities;
- documentation of the contingency measures concerning access control;
- information of the personnel about applicable practises in extreme cases. protection (e.g. for performing backups);

A.5.3 Example of a checklist

The examples of checklists for site visits consist in tables for the preparation of an audit and for the presentation of the results of an audit.

The checklist structure given in the following is preliminary. Dependent on the concrete contents of the new guideline, changes can become necessary.

The checklist is divided into three subclauses according to the subjects indicated in the introduction (A.4.1):

a) configuration management system;

b) delivery procedures;

c) security measures during development.

These subclauses correspond to the actual CC Part 3 class ALC, especially the families CM capabilities (ALC_CMC).n with n>=3, Delivery (ALC_DEL) and Development security (ALC_DVS).

The subclauses are subdivided further into rows corresponding to the relevant work units of this document.

The columns of the checklist contain in turn:

a) a consecutive number;

b) the referenced work unit;

c) the references to the corresponding developer documentation;

d) the explicit reproduction of the developer measures;

e) special remarks and questions to be clarified on the visit (beyond the standard evaluator task to verify the application of the indicated measures);

f) the result of the examinations during the visit.

If it is decided to have separate checklists for preparation and reporting of the audit, the result column is omitted in the preparation list and the remarks and questions column is omitted in the reporting list. The remaining columns should be identical in both lists.

Table A.1 — Example of a checklist at EAL 4 (extract)

A. Examination of the CM system (ALC_CMC.4 and ALC_CMS.4)					
No.	Work Unit	Developer Documentation	Measures	Questions and Remarks	Result
A.1	ALC_CMC.4-11 ALC_CMC.4-12	"Configuration Management System", ch. ...	The system automatically managing the source code files is capable of administering user profiles and graded access rights, and of checking identification and authentication of users.	Does reading or updating of a source code file require a user authentication?	If a user has not the right to access a confidential document, it is not even displayed in the file list.
…	…	…	…	…	…
B. Examination of the Delivery Procedures (ALC_DEL.1)					
No.	Work Unit	Developer Documentation	Measures	Questions and Remarks	Result
B.1	ALC_DEL.1-1, ALC_DEL.1-2	"Delivery of the TOE", ch. ...	The software is transmitted PGP-signed and encrypted to the customer.	---	The evaluators have checked the process and found it as described, additionally a checksum is transmitted.
…	…	…	…	…	…
C. Examination of the organisational and infrastructural developer security (ALC_DVS.1, ALC_LCD.1, ALC_TAT.1)					
No.	Work Unit	Developer Documentation	Measures	Questions and Remarks	Result

C.1	ALC_DVS.1-1, ALC_DVS.1-2	"Security of the development environment", ch. ... (Premises)	The premises are protected by security fencing.	Is the fencing sufficiently strong and high to prevent an easy intrusion into the premises?	The evaluators considered the fencing to be sufficiently strong and high.
C.2	ALC_DVS.1-1, ALC_DVS.1-2	"Security of the development environment", ch. ... (Building)	The building has the following access possibilities: The main entrance which is surveyed by the reception and is closed if the reception is not manned. And an access in the goods reception which is secured by two roller shutters.	Is the listing of the access possibilities complete?	Beyond the indicated access possibilities, there is an emergency exit that cannot be opened from the outside. The roller shutters mentioned before can be operated only from inside.
…	…	…	…	…	…

A.6 Scheme responsibilities

This document describes the minimum technical work that evaluations conducted under oversight (scheme) bodies must perform before issuing an oversight verdict. 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 this document 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 this document should consult with their evaluation schemes to determine under whose auspices the subject falls.)

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 technical competence, understanding of work and 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 document 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 evaluation authority 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 B
(informative)

Vulnerability assessment (AVA)

This annex provides an explanation of the AVA_VAN criteria and examples of their application. This annex does not define the AVA criteria; this definition can be found in CC Part 3, Class AVA: Vulnerability assessment.

This annex consists of two major parts:

a) guidance for completing an independent vulnerability analysis. This is summarized in B.1.1 and described in more detail in B.1.2. These subclauses describe how an evaluator should approach the construction of an independent vulnerability analysis.

b) how to characterise and use assumed Attack Potential of an attacker. This is described in B.1.5 to B.3. These subclauses provide an example of how an attack potential can be characterised and should be used, and provide examples.

B.1 What is vulnerability analysis

The purpose of the vulnerability assessment activity is to determine the existence and exploitability of flaws or weaknesses in the TOE in the operational environment. This determination is based upon analysis performed by the evaluator, and is supported by evaluator testing.

At the lowest levels of vulnerability analysis (AVA_VAN) the evaluator simply performs a search of publicly available information to identify any known weaknesses in the TOE, while at the higher levels the evaluator performs a structured analysis of the TOE evaluation evidence.

There are three main factors in performing a vulnerability analysis, namely:

a) the identification of potential vulnerabilities;

b) assessment to determine whether the identified potential vulnerabilities can allow an attacker with the relevant attack potential to violate the SFRs.

c) penetration testing to determine whether the identified potential vulnerabilities are exploitable in the operational environment of the TOE.

The identification of vulnerabilities can be further decomposed into the evidence to be searched and how hard to search that evidence to identify potential vulnerabilities. In a similar manner, the penetration testing can be further decomposed into analysis of the potential vulnerability to identify attack methods and the demonstration of the attack methods.

These main factors are iterative in nature, i.e. penetration testing of potential vulnerabilities may lead to the identification of further potential vulnerabilities. Hence, these are performed as a single vulnerability analysis activity.

B.2 Evaluator construction of a vulnerability analysis

The evaluator vulnerability analysis is to determine that the TOE is resistant to penetration attacks performed by an attacker possessing a Basic (for AVA_VAN.1 and AVA_VAN.2), Enhanced-

Basic (for AVA_VAN.3), Moderate (for AVA_VAN.4) or High (for AVA_VAN.5) attack potential. The evaluator first assesses the exploitability of all identified potential vulnerabilities. This is accomplished by conducting penetration testing. The evaluator should assume the role of an attacker with a Basic (for AVA_VAN.1 and AVA_VAN.2), Enhanced-Basic (for AVA_VAN.3), Moderate (for AVA_VAN.4) or High (for AVA_VAN.5) attack potential when attempting to penetrate the TOE.

The evaluator considers potential vulnerabilities encountered by the evaluator during the conduct of other evaluation activities. The evaluator penetration testing determining TOE resistance to these potential vulnerabilities should be performed assuming the role of an attacker with a Basic (for AVA_VAN.1 and AVA_VAN.2), Enhanced-Basic (for AVA_VAN.3), Moderate (for AVA_VAN.4) or High (for AVA_VAN.5) attack potential.

However, vulnerability analysis should not be performed as an isolated activity. It is closely linked with ADV and AGD. The evaluator performs these other evaluation activities with a focus on identifying potential vulnerabilities or "areas of concern". Therefore, evaluator familiarity with the generic vulnerability guidance (provided in B.1.3) is required.

B.3 Generic vulnerability guidance

The following five categories provide discussion of generic vulnerabilities.

B.3.1 Bypassing

Bypassing includes any means by which an attacker can 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.

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 TSFI;

    2) invoking an additional TSFI;

    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 invoke a TSFI (e.g. opening a file for access and then reading data from it). If a TSFI is invoked through one of the TOE interfaces (e.g. an access control check), the evaluator should

consider whether it is possible to bypass the control 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 described above, 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 the TSF 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 (see B.1.3.4 for further discussion of covert channels). The use of undocumented interfaces, which may be insecure, also falls into this category. Such interfaces may include undocumented support and help facilities;

e) using implementation detail introduced in lower representations may allow an attacker to take advantage of additional functions, resources or attributes that are introduced to the TOE as a consequence of the refinement process. Additional functionality may include test harness code contained in software modules and back-doors introduced during the implementation process;

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 can take advantage of. For example, if the TSF can 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 TSF of an 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 against an application on an underlying database management system: again the TSF can 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 disk 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.

There are a number of different methods through which an evaluator can identify a back-door, including two main techniques. Firstly, by the evaluator inadvertently identifying during testing an interface that can be misused. Secondly, through testing each external interface of the TSF in a debugging mode to identify any modules that are not called as a part of testing the documented interfaces and then inspecting the code that is not called to consider whether it is a back-door.

For a software TOE where evaluation of sub-activity (ADV_IMP.2) and ALC_TAT.2 or higher components are included in the assurance package, the evaluator can consider during their analysis of the tools the libraries and packages that are linked by the compiler at compilation stage to determine that back-doors are not introduced at this stage.

B.3.2 Tampering

Tampering includes any attack based on an attacker attempting to influence the behaviour of the TSF (i.e. corruption or de-activation), for example by:

a) accessing data on whose confidentiality or integrity the TSF relies;

b) forcing the TOE to cope with unusual or unexpected circumstances;

c) disabling or delaying security enforcement;

d) physically modifying the TOE.

Each of the following should be considered (where relevant) in the evaluator's independent vulnerability analysis:

a) attacks based on accessing data, whose confidentiality or integrity are protected, 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 interfaces 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 deceive 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 interfaces 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 and supplying improperly formatted input that should fail parsing such as SQL-injection, format strings);

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 can confer advantage to an attacker;

j) the correct behaviour of the TSF 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 disks;

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 interfaces 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 can be put into a resource-wait state if two other processes are accessing a resource which it requires;

o) using interfaces between components which are not visible at a higher level of abstraction may provide a means of delaying a time-critical trusted process;

p) physical attacks can be categorised into physical probing, physical manipulation, physical modification, and substitution:

    1) physical probing by penetrating the TOE targeting internals of the TOE, e.g. reading at internal communication interfaces, lines or memories;

    2) physical manipulation can be with the TOE internals aiming at internal modifications of the TOE (e.g. by using optical fault induction as an interaction process), at the external interfaces of the TOE (e.g. by power or clock glitches) and at the TOE environment (e.g. by modifying temperature);

    3) physical modification of TOE internal security enforcing attributes to inherit privileges or other capabilities that should be denied in regular operation. Such modifications can be caused, e.g., by optical fault induction. Attacks based on physical modification may also yield a modification of the TSF itself, e.g. by causing faults at TOE internal program data transfers before execution. Note that such kind of bypassing by modifying the TSF itself can jeopardise every TSF unless there are other measures (possibly environmental measures) that prevent an attacker from gaining physical access to the TOE;

    4) physical substitution to replace the TOE with another IT entity, during delivery or operation of the TOE. Substitution during delivery of the TOE from the development environment to the user should be prevented through application of secure delivery procedures, such as those considered under Development security (ALC_DVS). Substitution of the TOE during operation may be considered through a combination of user guidance and the operational environment, such that the user is able to be confident that they are interacting with the TOE.

B.3.3 Direct attacks

Direct attack includes the identification of any penetration tests necessary to test the strength of permutational or probabilistic mechanism and other mechanisms to ensure they withstand direct attack.

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.

Where a probabilistic or permutational mechanism relies on selection of security attribute value (e.g. selection of password length) or entry of data by a human user (e.g. choice of password), the assumptions made should reflect the worst case.

Probabilistic or permutational mechanisms should be identified during examination of evaluation evidence required as input to this sub-activity (security target, functional specification, TOE design and implementation representation subset) and any other TOE (e.g. guidance) documentation may identify additional probabilistic or permutational mechanisms.

Where the design evidence or guidance 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 can be achieved through testing or through independent analysis.

Direct attacks reliant upon a weakness in a cryptographic algorithm should not be considered under vulnerability analysis (AVA_VAN), as this is outside the scope of CC Part 3. Correctness of the implementation of the cryptographic algorithm is considered during the ADV and ATE activities.

B.3.4 Monitoring

Information is an abstract view on relation between the properties of entities, i.e. a signal contains information for a system, if the TOE is able to react to this signal. The TOE resources processes and stores information represented by user data. Therefore:

a) information may flow with the user data between subjects by internal TOE transfer or export from the TOE;

b) information may be generated and passed to other user data;

c) information may be gained through monitoring the operations on data representing the information.

The information represented by user data may be characterised by security attributes like "classification level" having values, for example unclassified, confidential, secret, top secret, to control operations to the data. This information and therefore the security attributes may be changed by operations e.g. FDP_ACC.2 may describe decrease of the level by "sanitisation" or increase of level by combination of data. This is one aspects of an information flow analysis focused on controlled operations of controlled subjects on controlled objects.

The other aspect is the analysis of ***illicit information flow***. This aspect is more general than the direct access to objects containing user data addressed by the FDP_ACC family. An ***unenforced*** signalling channel carrying information under control of the information flow control policy can also be caused by monitoring of the processing of any object containing or related to this information (e.g. side channels). An ***enforced*** signalling channel may be identified in terms of the subjects manipulating resources and the subject or user that observe such manipulation. Classically, covert channels have been identified as timing or storage channels, according to the resource being modified or modulated. As for other monitoring attacks, the use of the TOE is in accordance with the SFRs.

Covert channels are normally applicable in the case when the TOE has unobservability and multi-level separation policy requirements. Covert channels can be routinely spotted during vulnerability analysis and design activities, and should therefore be tested. However, generally such monitoring attacks are only identified through specialised analysis techniques commonly referred to as "covert channel analysis". These techniques have been the subject of much research and there are many papers published on this subject. Guidance for the conduct of covert channel analysis should be sought from the evaluation authority.

Unenforced information flow monitoring attacks include passive analysis techniques aiming at disclosure of sensitive internal data of the TOE by operating the TOE in the way that corresponds to the guidance documents.

Side Channel Analysis includes crypt analytical techniques based on physical leakage of the TOE. Physical leakage can occur by timing information, power consumption or power emanation during computation of a TSF. Timing information can be collected also by a remote-attacker (having network access to the TOE), power based information channels require that the attacker is in the near-by environment of the TOE.

Eavesdropping techniques include interception of all forms of energy, e.g. electromagnetic or optical emanation of computer displays, not necessarily in the near-field of the TOE.

Monitoring also includes exploits of protocol flaws, e.g., an attack on SSL implementation.

B.3.5 Misuse

Misuse can arise from:

a) incomplete guidance documentation;

b) unreasonable guidance;

c) unintended misconfiguration of the TOE;

d) forced exception behaviour of the TOE.

If the guidance documentation is incomplete the user may not know how to operate the TOE in accordance with the SFRs. The evaluator should apply familiarity with the TOE gained from performing other evaluation activities to determine that the guidance is complete. In particular, the evaluator should consider the functional specification. The TSF described in this document should be described in the guidance as required to permit secure administration and use through the TSFI available to human users. In addition, the different modes of operation should be considered to ensure that guidance is provided for all modes of operation.

The evaluator can, as an aid, prepare an informal mapping between the guidance and these documents. Any omissions in this mapping can indicate incompleteness.

The guidance is considered to be 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.

A TOE may use a variety of ways to assist the consumer in effectively using that TOE in accordance with the SFRs and prevent unintentional misconfiguration. A TOE may employ functionality (features) to alert the consumer when the TOE is in a state that is inconsistent with the SFRs, 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 when the SFRs are being compromised; namely insecure.

The evaluator considers the TOE's functionality, its purpose and security objectives for the operational environment to arrive at a conclusion of whether or not there is reasonable expectation that use of the guidance would permit transition into an insecure state to be detected in a timely manner.

The potential for the TOE to enter into insecure states can be determined using the evaluation deliverables, such as the ST, the functional specification and any other design representations provided as evidence for components included in the assurance package for the TOE (e.g. the TOE/TSF design specification if a component from TOE design (ADV_TDS) is included).

Instances of forced exception behaviour of the TSF can include, but are not limited to, the following:

a) behaviour of the TOE when start-up, close-down or error recovery is activated;

b) behaviour of the TOE under extreme circumstances (sometimes termed overload or asymptotic behaviour), particularly where this can lead to the de-activation or disabling of parts of the TSF;

c) any potential for unintentional misconfiguration or insecure use arising from attacks noted in the subclause on tampering above.

B.4 Identification of potential vulnerabilities

Potential vulnerabilities can be identified by the evaluator during different activities. They can become apparent during an evaluation activity or they can be identified as a result of analysis of evidence to search for vulnerabilities.

B.4.1 Encountered

The encountered identification of vulnerabilities is where potential vulnerabilities are identified by the evaluator during the conduct of evaluation activities, i.e. the evidence are not being analysed with the express aim of identifying potential vulnerabilities.

The encountered method of identification is dependent on the evaluator's experience and knowledge; which is monitored and controlled by the evaluation authority. It is not reproducible in approach but will be documented to ensure repeatability of the conclusions from the reported potential vulnerabilities.

There are no formal analysis criteria required for this method. Potential vulnerabilities are identified from the evidence provided as a result of knowledge and experience. However, this method of identification is not constrained to any particular subset of evidence.

Evaluator is assumed to have knowledge of the TOE-type technology and known security flaws as documented in the public domain. The level of knowledge assumed is that which can be gained from a security e-mail list relevant to the TOE type, the regular bulletins (bug, vulnerability and security flaw lists) published by those organisations researching security issues in products and technologies in widespread use. This knowledge is not expected to extend to specific conference proceedings or detailed theses produced by university research for AVA_VAN.1 or AVA_VAN.2. However, to ensure the knowledge applied is up to date, the evaluator can need to perform a search of public domain material.

For AVA_VAN.3 to AVA_VAN.5 the search of publicly available information is expected to include conference proceeding and theses produced during research activities by universities and other relevant organisations.

Examples of how these can arise (how the evaluator can encounter potential vulnerabilities):

a) while the evaluator is examining some evidence, it sparks a memory of a potential vulnerability identified in a similar product type, that the evaluator believes to also be present in the TOE under evaluation;

b) while examining some evidence, the evaluator spots a flaw in the specification of an interface, that reflects a potential vulnerability.

This can include becoming aware of a potential vulnerability in a TOE through reading about generic vulnerabilities in a particular product type in an IT security publication or on a security e-mail list to which the evaluator is subscribed.

Attack methods can be developed directly from these potential vulnerabilities. Therefore, the encountered potential vulnerabilities are collated at the time of producing penetration tests based on the evaluator's vulnerability analysis. There is no explicit action for the evaluator to encounter potential vulnerabilities. Therefore, the evaluator is directed through an implicit action specified in AVA_VAN.1.2E and AVA_VAN.*.4E.

Current information regarding public domain vulnerabilities and attacks can be provided to the evaluator by, for example, an evaluation authority. This information is to be taken into account by the evaluator when collating encountered vulnerabilities and attack methods when developing penetration tests.

B.4.2 Analysis

B.4.2.1 General

The following types of analysis are presented in terms of the evaluator actions.

B.4.2.2 Unstructured analysis

The unstructured analysis to be performed by the evaluator [for Evaluation of sub-activity (AVA_VAN.2)] permits the evaluator to consider the generic vulnerabilities (as discussed in B.1.3). The evaluator will also apply their experience and knowledge of flaws in similar technology types.

B.4.2.3 Focused

During the conduct of evaluation activities, the evaluator can also identify areas of concern. These are specific portions of the TOE evidence that the evaluator has some reservation about, although the evidence meets the requirements for the activity with which the evidence is associated. For example, a particular interface specification looks particularly complex, and therefore may be prone to error either in the development of the TOE or in the operation of the TOE. There is no potential vulnerability apparent at this stage, further investigation is required. This is beyond the bounds of encountered, as further investigation is required.

Difference between potential vulnerability and area of concern:

a) potential vulnerability - The evaluator knows a method of attack that can be used to exploit the weakness or the evaluator knows of vulnerability information that is relevant to the TOE;

b) area of concern - The evaluator may be able to discount concern as a potential vulnerability based on information provided elsewhere. While reading interface specification, the evaluator identifies that due to the extreme (unnecessary) complexity of an interface a potential vulnerability may lay within that area, although it is not apparent through this initial examination.

The focused approach to the identification of vulnerabilities is an analysis of the evidence with the aim of identifying any potential vulnerabilities evident through the contained information. It is an unstructured analysis, as the approach is not predetermined. This approach to the identification of potential vulnerabilities can be used during the independent vulnerability analysis required by evaluation of sub-activity (AVA_VAN.3).

This analysis can be achieved through different approaches, that will lead to commensurate levels of confidence. None of the approaches have a rigid format for the examination of evidence to be performed.

The approach taken is directed by the results of the evaluator's assessment of the evidence to determine it meets the requirements of the AVA/AGD sub-activities. Therefore, the investigation

of the evidence for the existence of potential vulnerabilities may be directed by any of the following:

a) areas of concern identified during examination of the evidence during the conduct of evaluation activities;

b) reliance on particular functionality to provide separation, identified during the analysis of the architectural design, e.g. Evaluation of sub-activity (ADV_ARC.1), requiring further analysis to determine it cannot be bypassed;

c) representative examination of the evidence to hypothesise potential vulnerabilities in the TOE.

The evaluator will report what actions were taken to identify potential vulnerabilities in the evidence. However, the evaluator may not be able to describe the steps in identifying potential vulnerabilities before the outset of the examination. The approach will evolve as a result of the outcome of evaluation activities.

The areas of concern may arise from examination of any of the evidence provided to satisfy the SARs specified for the TOE evaluation. The information publicly accessible is also considered.

The activities performed by the evaluator can be repeated and the same conclusions, in terms of the level of assurance in the TOE, can be reached although the steps taken to achieve those conclusions may vary. As the evaluator is documenting the form the analysis took, the actual steps taken to achieve those conclusions are also reproducible.

B.4.2.4 Methodical

The methodical analysis approach takes the form of a structured examination of the evidence. This method requires the evaluator to specify the structure and form the analysis will take (i.e. the manner in which the analysis is performed is predetermined, unlike the focused identification method). The method is specified in terms of the information that will be considered and how/why it will be considered. This approach to the identification of potential vulnerabilities can be used during the independent vulnerability analysis required by evaluation of sub-activity (AVA_VAN.4) and evaluation of sub-activity (AVA_VAN.5).

This analysis of the evidence is deliberate and pre-planned in approach, considering all evidence identified as an input into the analysis.

All evidence provided to satisfy the (ADV) assurance requirements specified in the assurance package are used as input to the potential vulnerability identification activity.

The "methodical" descriptor for this analysis has been used in an attempt to capture the characterisation that this identification of potential vulnerabilities is to take an ordered and planned approach. A "method" or "system" is to be applied in the examination. The evaluator is to describe the method to be used in terms of what evidence will be considered, the information within the evidence that is to be examined, the manner in which this information is to be considered; and the hypothesis that is to be generated.

The following provide some examples that a hypothesis may take:

a) consideration of malformed input for interfaces available to an attacker at the external interfaces;

b) examination of a security mechanism, such as domain separation, hypothesising internal buffer overflows leading to degradation of separation;

c) analysis to identify any objects created in the TOE implementation representation that are then not fully controlled by the TSF, and can be used by an attacker to undermine the SFRs.

For example, the evaluator may identify that interfaces are a potential area of weakness in the TOE and specify an approach to the analysis that "all interface specifications provided in the functional specification and TOE design will be analysed to hypothesise potential vulnerabilities" and go on to explain the methods used in the hypothesis.

This identification method will provide a plan of attack of the TOE, that would be performed by an evaluator completing penetration testing of potential vulnerabilities in the TOE. The rationale for the method of identification would provide the evidence for the coverage and depth of exploitation determination that would be performed on the TOE.

B.5 When attack potential is used

B.5.1 Developer

Attack potential is used by a PP/ST author during the development of the PP/ST, in consideration of the threat environment and the selection of assurance components. This may simply be a determination that the attack potential possessed by the assumed attackers of the TOE is generically characterised as Basic, Enhanced-Basic, Moderate or High. Alternatively, the PP/ST may wish to specify particular levels of individual factors assumed to be possessed by attackers. (e.g. the attackers are assumed to be experts in the TOE technology type, with access to specialised equipment.)

The PP/ST author considers the threat profile developed during a risk assessment (outside the scope of the CC, but used as an input into the development of the PP/ST in terms of the Security Problem Definition or in the case of Direct Rationale STs, the requirements statement). Consideration of this threat profile in terms of one of the approaches discussed in the following subclauses will permit the specification of the attack potential the TOE is to resist.

B.5.2 Evaluator

Attack potential is especially considered by the evaluator in two distinct ways during the ST evaluation and the vulnerability assessment activities.

Attack potential is used by an evaluator during the conduct of the vulnerability analysis sub-activity to determine whether or not the TOE is resistant to attacks assuming a specific attack potential of an attacker. If the evaluator determines that a potential vulnerability is exploitable in the TOE, they have to confirm that it is exploitable considering all aspects of the intended environment, including the attack potential assumed by an attacker.

Therefore, using the information provided in the threat statement of the Security Target, 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 demonstrates the relationship between this analysis and attack potential.

Table B.1 — Vulnerability testing and attack potential

Vulnerability Component	TOE resistant to attacker with attack potential of:	Residual vulnerabilities only exploitable by attacker with attack potential of:
VAN.5	High	Beyond High
VAN.4	Moderate	High

Vulnerability Component	TOE resistant to attacker with attack potential of:	Residual vulnerabilities only exploitable by attacker with attack potential of:
VAN.3	Enhanced-Basic	Moderate
VAN.2	Basic	Enhanced-Basic
VAN.1	Basic	Enhanced-Basic

The "beyond high" entry in the residual vulnerabilities column of the above table represents those potential vulnerabilities that would require an attacker to have an attack potential greater than that of "high" in order to exploit the potential vulnerability. A vulnerability classified as residual in this instance reflects the fact that a known weakness exists in the TOE, but in the current operational environment, with the assumed attack potential, the weakness cannot be exploited.

At any level of attack potential a potential vulnerability may be deemed "infeasible" due to a countermeasure in the operational environment that prevents the vulnerability from being exploited.

A vulnerability analysis applies to all TSFI, including ones that access probabilistic or permutational mechanisms. No assumptions are made regarding the correctness of the design and implementation of the TSFI; 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 that the TSF is designed and implemented to protect against the required level of threat.

It is not necessary for an evaluator to perform an attack potential calculation for each potential vulnerability. In some cases, it is apparent when developing the attack method whether or not the attack potential required to develop and run the attack method is commensurate with that assumed of the attacker in the operational environment. For any vulnerabilities for which an exploitation is determined, the evaluator performs an attack potential calculation to determine that the exploitation is appropriate to the level of attack potential assumed for the attacker.

The approach described below is to be applied whenever it is necessary to calculate attack potential, unless the evaluation authority provides mandatory guidance that an alternative approach is to be applied. The values given in Tables B.2 and B.3 below are not mathematically proven. Therefore, the values given in these example tables may need to be adjusted according to the technology type and specific environments. The evaluator should seek guidance from the evaluation authority.

B.6 Calculating attack potential

B.6.1 Application of attack potential

B.6.1.1 General

Attack potential is a function of expertise, resources and motivation. There are multiple methods of representing and quantifying these factors. Also, there may be other factors that are applicable for particular TOE types.

B.6.1.2 Treatment of motivation

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.

Secondly, motivation can imply the value of the asset, monetarily or otherwise, to 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.

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.

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 operational 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.

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 operational environment, is sufficient to thwart attackers with the identified expertise and resources.

It may be possible for a PP author to quantify the motivation of an attacker, as the PP author has greater knowledge of the operational environment in which the TOE (conforming to the requirements of the PP) is to be placed. Therefore, the motivation can form an explicit part of the expression of the attack potential in the PP, along with the necessary methods and measures to quantify the motivation.

B.6.2 Characterising attack potential

B.6.2.1 General

This subclause 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.

B.6.2.2 Determining the attack potential

The determination of the attack potential for an attack corresponds to the identification of the effort required to create the attack, and to demonstrate that it can be successfully applied to the TOE (including setting up or building any necessary test equipment), thereby exploiting the vulnerability in the TOE. The demonstration that the attack can be successfully applied needs to consider any difficulties in expanding a result to create a useful attack. For example, where an experiment reveals some bits or bytes of a confidential data item (such as a key), it is necessary to consider how the remainder of the data item would be obtained (in this example some bits can

be measured directly by further experiments, while others can be found by a different technique such as exhaustive search). It may not be necessary to carry out all of the experiments to identify the full attack, provided it is clear that the attack actually proves that access has been gained to a TOE asset, and that the complete attack can realistically be carried out in exploitation according to the AVA_VAN component targeted. In some cases, the only way to prove that an attack can realistically be carried out in exploitation according to the AVA_VAN component targeted is to perform completely the attack and then rate it based upon the resources actually required. One of the outputs from the identification of a potential vulnerability is assumed to be a script that gives a step-by-step description of how to carry out the attack that can be used in the exploitation of the vulnerability on another instance of the TOE.

In many cases, the evaluators will estimate the parameters for exploitation, rather than carry out the full exploitation. The estimates and their rationale will be documented in the ETR.

B.6.2.3 Factors to be considered

The following factors should be considered during analysis of the attack potential required to exploit a vulnerability:

a) Time taken to identify and exploit (***Elapsed Time***);

b) Specialist technical expertise required (***Specialist Expertise***);

c) Knowledge of the TOE design and operation (***Knowledge of the TOE***);

d) Window of opportunity;

e) IT hardware/software or other equipment required for exploitation.

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. (The levels of each factor are discussed in increasing order of magnitude.) When it is the case, the less "expensive" combination is considered in the exploitation phase.

Elapsed time is the total amount of time taken by an attacker to identify that a particular potential vulnerability may exist in the TOE, to develop an attack method and to sustain effort required to mount the attack against the TOE. When considering this factor, the worst-case scenario is used to estimate the amount of time required. The identified amount of time is as follows:

a) less than one day;

b) between one day and one week;

c) between one week and two weeks;

d) between two weeks and one month;

e) each additional month up to six months leads to an increased value;

f) more than six months.

Specialist expertise refers to the level of generic knowledge of the underlying principles, product type or attack methods (e.g. Internet protocols, Unix operating systems, buffer overflows). The identified levels are as follows:

a) Laymen are unknowledgeable compared to experts or proficient persons, with no particular expertise;

b) Proficient persons are knowledgeable in that they are familiar with the security behaviour of the product or system type;

c) Experts are familiar with the underlying algorithms, protocols, hardware, structures, security behaviour, principles and concepts of security employed, techniques and tools for the definition of new attacks, cryptography, classical attacks for the product type, attack methods, etc. implemented in the product or system type.

The level "Multiple Expert" is introduced to allow for a situation, where different fields of expertise are required at an Expert level for distinct steps of an attack.

It may occur that several types of expertise are required. By default, the higher of the different expertise factors is chosen. In very specific cases, the "multiple expert" level can be used but it should be noted that the expertise must concern fields that are strictly different like for example HW manipulation and cryptography.

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) Public information concerning the TOE (e.g. as gained from the Internet);

b) Restricted information concerning the TOE (e.g. knowledge that is controlled within the developer organisation and shared with other organisations under a non-disclosure agreement);

c) Sensitive information about the TOE (e.g. knowledge that is shared between discreet teams within the developer organisation, access to which is constrained only to members of the specified teams);

d) Critical information about the TOE (e.g. knowledge that is known by only a few individuals, access to which is very tightly controlled on a strict need to know basis and individual undertaking).

The knowledge of the TOE may graduate according to design abstraction, although this can only be done on a TOE by TOE basis. Some TOE designs may be public source (or heavily based on public source) and therefore even the design representation would be classified as public or at most restricted, while the implementation representation for other TOEs is very closely controlled as it would give an attacker information that would aid an attack and is therefore considered to be sensitive or even critical.

It may occur that several types of knowledge are required. In such cases, the higher of the different knowledge factors is chosen.

Window of opportunity (Opportunity) is also an important consideration, and has a relationship to the ***Elapsed 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 attack methods 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 some TOEs the ***Window of opportunity*** may equate to the number of samples of the TOE that the attacker can obtain. This is particularly relevant where attempts to penetrate the TOE and undermine the SFRs may result in the destruction of the TOE preventing use of that TOE sample for further testing, e.g. hardware devices. Often in these cases distribution of the TOE is controlled and so the attacker must apply effort to obtain further samples of the TOE.

For the purposes of this discussion:

a) unnecessary/unlimited access means that the attack doesn't need any kind of opportunity to be realised because there is no risk of being detected during access to the TOE and it is no problem to access the number of TOE samples for the attack;

b) easy means that access is required for less than a day and that the number of TOE samples required to perform the attack is less than ten;

c) moderate means that access is required for less than a month and that the number of TOE samples required to perform the attack is less than one hundred;

d) difficult means that access is required for at least a month or that the number of TOE samples required to perform the attack is at least one hundred;

e) none means that the opportunity window is not sufficient to perform the attack (the length for which the asset to be exploited is available or is sensitive is less than the opportunity length needed to perform the attack - for example, if the asset key is changed each week and the attack needs two weeks); another case is, that a sufficient number of TOE samples needed to perform the attack is not accessible to the attacker - for example if the TOE is a hardware and the probability to destroy the TOE during the attack instead of being successful is very high and the attacker has only access to one sample of the TOE.

Consideration of this factor can result in determining that it is not possible to complete the exploit, due to requirements for time availability that are greater than the opportunity time.

IT hardware/software or other equipment refers to the equipment required to identify or exploit a vulnerability.

Standard equipment 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, protocol analyser or simple attack scripts).

Specialised equipment is not readily available to the attacker, but can be acquired without undue effort. This can include purchase of moderate amounts of equipment (e.g. power analysis tools, use of hundreds of PCs linked across the Internet would fall into this category), or development of more extensive attack scripts or programs. If clearly different test benches consisting of specialised equipment are required for distinct steps of an attack this would be rated as bespoke.

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.

The level "Multiple Bespoke" is introduced to allow for a situation, where different types of bespoke equipment are required for distinct steps of an attack.

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 (where the attacker may be one or more persons with complementary areas of knowledge) 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 (IT hardware/software or other 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 be effectively used are created and freely distributed (e.g. via the Internet).

B.6.2.4 Calculation of attack potential

Table B.2 identifies the factors discussed in the previous subclause and associates numeric values with the total value of each factor.

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 twenty samples are required to perform the attack then a value between one and four may be selected for that factor, or if the design is based on a publicly available design but the developer has made some alterations then a value between zero and three should be selected according to the evaluator's view of the impact of those design changes. The table is intended as a guide.

The "**" specification in the table in considering ***Window of Opportunity*** is not to be seen as a natural progression from the timescales specified in the preceding ranges associated with this factor. This specification identifies that for a particular reason the potential vulnerability cannot be exploited in the TOE in its intended operational environment. For example, access to the TOE may be detected after a certain amount of time in a TOE with a known environment (i.e. in the case of a system) where regular patrols are completed, and the attacker cannot gain access to the TOE for the required two weeks undetected. However, this would not be applicable to a TOE connected to the network where remote access is possible, or where the physical environment of the TOE is unknown.

Table B.2 — Calculation of attack potential

Factor	**Value**
Elapsed Time	
<= one day	0
<= one week	1
<= two weeks	2
<= one month	4
<= two months	7
<= three months	10
<= four months	13
<= five months	15
<= six months	17
> six months	19
Expertise	
Layman	0
Proficient	3[a]
Expert	6
Multiple experts	8
Knowledge of TOE	
Public	0
Restricted	3
Sensitive	7
Critical	11
Window of Opportunity	
Unnecessary / unlimited access	0
Easy	1
Moderate	4
Difficult	10
None	**[b]
Equipment	

Vulnerability assessment

Factor	Value
Elapsed Time	
Standard	0
Specialised	4[c]
Bespoke	7
Multiple bespoke	9

[a] When several proficient persons are required to complete the attack path, the resulting level of expertise still remains "proficient" (which leads to a 3 rating).

[b] Indicates that the attack path is not exploitable due to other measures in the intended operational environment of the TOE.

[c] If clearly different test benches consisting of specialised equipment are required for distinct steps of an attack, this should be rated as bespoke.

To determine the resistance of the TOE to the potential vulnerabilities identified the following steps should be applied:

a) Define the possible attack scenarios {AS1, AS2, ..., ASn} for the TOE in the operational environment.

b) For each attack scenario, perform a theoretical analysis and calculate the relevant attack potential using Table B.2.

For each attack scenario, if necessary, perform penetration tests in order to confirm or to disprove the theoretical analysis.

Divide all attack scenarios {AS1, AS2, ..., ASn} into two groups:

a) the attack scenarios having been successful (i.e. those that have been used to successfully undermine the SFRs), and

b) the attack scenarios that have been demonstrated to be unsuccessful.

For each successful attack scenario, apply Table B.3 and determine, whether there is a contradiction between the resistance of the TOE and the chosen AVA_VAN assurance component, see the last column of Table B.3.

Should one contradiction be found, the vulnerability assessment will fail, e.g. the author of the ST chose the component AVA_VAN.5 and an attack scenario with an attack potential of 21 points (high) has broken the security of the TOE. In this case, the TOE is resistant to attacker with attack potential 'Moderate', this contradicts to AVA_VAN.5, hence, the vulnerability assessment fails.

The "Values" column of Table B.3 indicates the range of attack potential values (calculated using Table B.2) of an attack scenario that results in the SFRs being undermined.

Table B.3 — Rating of vulnerabilities and TOE resistance

Values	Attack potential required to exploit scenario:	Meets assurance components:	Failure of components:
0-9	Basic	-	AVA_VAN.1, AVA_VAN.2, AVA_VAN.3, AVA_VAN.4, AVA_VAN.5
10-13	Enhanced-Basic	AVA_VAN.1, AVA_VAN.2	AVA_VAN.3, AVA_VAN.4, AVA_VAN.5
14-19	Moderate	AVA_VAN.1, AVA_VAN.2, AVA_VAN.3	AVA_VAN.4, AVA_VAN.5
20-24	High	AVA_VAN.1, AVA_VAN.2, AVA_VAN.3, AVA_VAN.4	AVA_VAN.5
=>25	Beyond High	AVA_VAN.1, AVA_VAN.2, AVA_VAN.3, AVA_VAN.4, AVA_VAN.5	-

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 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 can indicate.

It should be noted that whereas a number of vulnerabilities rated individually may indicate high resistance to attack, collectively the combination of vulnerabilities may indicate that overall a lower rating is applicable. The presence of one vulnerability may make another easier to exploit.

If a PP/ST author wants to use the attack potential table for the determination of the level of attack the TOE should withstand (selection of Vulnerability analysis (AVA_VAN) component), they should proceed as follows: For all different attack scenarios (i.e. for all different types of attacker and/or different types of attack the author has in mind) which must not violate the SFRs, several passes through Table B.2 should be made to determine the different values of attack potential assumed for each such unsuccessful attack scenario. The PP/ST author then chooses the highest value of them in order to determine the level of the TOE resistance to be claimed from Table B.3: the TOE resistance must be at least equal to this highest value determined. For example, the highest value of attack potentials of all attack scenarios, which must not undermine the TOE security policy, determined in such a way is Moderate; hence, the TOE resistance shall be at least Moderate (i.e. Moderate or High); therefore, the PP/ST author can choose either AVA_VAN.4 (for Moderate) or AVA_VAN.5 (for High) as the appropriate assurance component.

B.7 Example calculation for direct attack

Mechanisms subject to direct attack are often vital for system security and developers often strengthen these mechanisms. As an example, a TOE can use a simple pass number authentication mechanism that can be overcome by an attacker who has the opportunity to repeatedly guess another user's pass number. The system can strengthen this mechanism by restricting pass numbers and their use in various ways. During the course of the evaluation an analysis of this direct attack can proceed as follows:

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. Authorized 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).

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.

The pass number space is calculated as follows:

Patterns of human usage are important considerations that can influence the approach to searching a password space. 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:

$$7(8)(9)(10) = 5040$$

The number of possible increasing sequences is seven, as is the number of decreasing sequences. The pass number space after disallowing sequences is:

$$5040 - 14 = 5026$$

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.

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:

$$\frac{2513 min}{60 \frac{min}{hour}} \approx 42 hours$$

Using the approach to calculate the attack potential, described in the previous subclause, identifies that it is possible that a layman can defeat the mechanism within days (given easy

access to the TOE), with the use of standard equipment, and with no knowledge of the TOE, giving a value of 1. Given the resulting sum, 1, the attack potential required to effect a successful attack is not rated, as it falls below that considered to be Basic.

Annex C
(informative)

Evaluation techniques and tools

C.1 Semiformal and formal methods

C.1.1 General

In CC Part 3, A.5, supplementary material on ADV_SPM and the relationship with the security target and the functional specification is provided.

C.1.2 Description of styles

This subclause provides general guidance on specification styles. Specific and detailed information is given in the work units corresponding to the specific evaluator action elements where examination of the style of specifications, (formal) TSF representations and correspondence demonstrations has to be performed.

The ADV class mandates three types of specification styles: informal, semiformal and formal. These styles are briefly described in the application notes to the ADV class in CC Part 3. The functional specification and design specification will be written using one or more of these specification styles. The TSF representations (in the following referred to as specifications) may use one or more notations in semiformal and formal style. The level of formality of the correspondence representation depends on the style of the adjacent pair of provided TSF representations (see the ADV_TDS family for details).

The hierarchy of components within these families increases the formality of the style:

- to reduce the ambiguity of the TSF representation;
- to reduce the likelihood of refinement errors in the available TSF representations;
- to strengthen the evidence for correctness of the TSF representations and the methods for their examination.

The styles are characterised by:

- informal style - defined semantics;
- semiformal style - defined semantics and syntax;
- formal style - defined semantics, syntax and rules of inference.

Regarding the notions of semantics and syntax, the degree of precision varies with the style of description.

Informal descriptions require the semantics to provide meaning to all terms that are used in a context other than that accepted by normal usage. No notational restrictions are imposed other than those required by the grammar and syntax conventions for the natural language which is used to provide the explanations.

Semiformal descriptions restrict the syntax formation of terms to well-defined expressions whose meaning has been precisely established either in an appendix to the description or in an external, public document used and listed as a reference.

Formal style descriptions restrict the semantics and syntax even further: the formation of syntactical terms follows a formal language description required to be decidable. Examples include well-established implicit formation rules being as precise as the formation of terms and formulas in first order predicate calculus or formal meta-language descriptions using Extended Backus-Naur Form (EBNF). Apart from informal descriptions the semantics of formal terms is restricted to well-established mathematical models. Formal derivation of theorems is restricted to predefined inference rules, which are based on well-known logical reasoning (classical logic, intuitionistic logic, modal logic, temporal logic, etc.).

In the context of formality levels, informal, semiformal, and formal styles are considered to be hierarchical in nature. Thus, requirements for an informal or semiformal style of specification may also be met with either a semiformal or formal specification style provided that it is supported by informal, explanatory text where appropriate. The set of presentation elements, syntactic and semantic rules is referred in the following as notation. A formal style of presentation uses a formal notation and rules of inference; in the following this is referred to as a formal system.

The content and presentation elements of ADV_FSP, ADV_SPM and ADV_TDS components describe the style in which the presentation of evidence shall be provided by the developer. The evaluator action element requires the evaluator to confirm that the information provided meets all requirements for presentation of evidence. If the content and presentation elements require an informal style the evaluator may perform the work units for the evaluator action elements in parallel with the other work units examining the content of evidence. If the content and presentation elements require a semiformal or a formal style this implies the application of semiformal or formal methods to examine the content. Therefore, it is recommended to perform the work units for the evaluator action elements concerning the correct use of the examination method and its necessary rigour before the analysis of the content of evidence. If a notation or its usage in the documentation does not provide the expected level of formality, then the necessary rigorous methods of analysis may not be applicable. The work unit for the evaluator action elements examining the necessary informal explanatory text may be performed in parallel with the other work units. Of course, the evaluator can detect errors in the presentation of evidence during the evaluator action as well which results in a fail verdict for the evaluator action elements.

The following text provides a guidance for the examination of specification styles and their use for correspondence demonstration in the sub-activities for the assurance families ADV_FSP, ADV_SPM and ADV_TDS.

C.1.2.1 Informal style

An informal specification is one that is expressed in a natural language. If content and presentation elements require an informal specification, the work unit for the evaluator action elements will require the evaluator to determine that it contains all necessary informal explanatory text. The evaluator should examine the specification to make sure that:

- it provides defined meanings of terms, abbreviations and acronyms that are used in a context other than that accepted by normal usage;
- if semiformal or formal notations are used, then appropriate informal, explanatory text supports the understanding.

This enforces the informal specification to provide defined semantics of its statements. An informal specification uses the ordinary conventions for the natural language i.e. any common spoken tongue. It may use figures and semiformal elements of presentation like data flow

diagrams to illustrate the informal specification. If the specification uses a semiformal notation it will be accompanied by supporting explanatory informal text appropriate for unambiguous common understanding.

Examples for the use of informal style are:

- CC Part 1 identifies a list of terms specific to the CC (all parts) and reserved terms in accordance with the ISO definitions contained in ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards and section 6 of this document. This clarifies the use of the verbs "shall", "should", "may" and "can" in the context of the CC (all parts);

- International standards and the Request for Interpretation (RFC) are specified in an informal style. They use semiformal notations as well, e.g. the abstract syntax notation ASN.1 for specification of message formats.

Informal style does not justify the absence of precision or informal definitions. The evaluator will assign a fail verdict if any of the following applies:

- some technical terms remain undefined;

- the lack of information prevents decision on the evaluator's side;

- ambiguous interpretations cause confusion.

C.1.2.2 Semiformal style

A semiformal specification is expressed in a restricted syntax language with defined semantics. It reduces the ambiguity of specification and strengthens the method of analysis.

The evaluator should examine the identified notations to make sure that:

- the syntax rules are defined or a definition is referenced;

- the notations with the explanatory text provide a defined semantics which is characterised by well-defined meanings of any terms, abbreviations and acronyms that are used in a context other than that accepted by normal usage;

- the use of a semiformal notation is accompanied by supporting explanatory text in informal style conducive to unambiguous meaning;

- the notations contain a restricted syntax language which means that a set of conventions must be supplied to define the restrictions imposed on the syntax.

Examples for the use of a semiformal style are:

- the restricted syntax language may be a natural language with restricted sentence structure and keywords with special meanings. CC Part 1 and CC Part 2 provide a semiformal notation for the security functional requirements consisting of classes, families and components together with rules for permitted operations. As required by the ECD families of classes ASE and APE, an explicitly stated IT security requirement shall use the requirements components, families and classes of the CC as a model for presentation;

- formally specified languages may be used to define the data structures for the use of TSFI or an interface of subsystems or modules in semiformal style. The interface specification may describe the complete details of all effects caused by interface usage by means of other semiformal notations e.g. state-transition diagrams;

- diagrams are commonly used for the specification of data-flow, state-transition, entity-relationship, data, process or program structures in a semiformal style. The graphical presentation assists the understanding of interaction and behaviour of entities depending on events. The abstraction accompanied by the graphical presentation normally needs to be compensated by informal description. Data-flow and state-transition diagrams may be very helpful, e.g. for the precise description and the analysis of protocols;

- programming languages define a strong syntax and well-defined semantics. The source code together with supporting explanatory text and documentation of well-defined development tools provide an unambiguous semiformal description of the TSF implementation, its security features and interfaces.

These examples show that the semiformal style covers a wide range of capabilities and levels of formality. The developer should use appropriate notation for the presentation of evidence depending on the type of the TOE (e.g. hardware, software), the development methodology and the purpose of the specification.

The semiformal style supports a structured analysis of the content, the consistency, the completeness and the correspondence of the representation. A semiformal analysis is one that results from a structured approach with a substantial degree of rigour in terms of completeness and correctness.

A semiformal interface specification supports the evaluator in analysing and assessing the external behaviour of a TSF, its subsystems or modules for any input (e.g. to decide regarding acceptance or rejection of a message and its content analysis). Semiformal evidence for conservation of properties can be obtained by means of flow charts and state transition diagrams visualizing the uniquely defined states and their interrelationship during the course of security preserving transitions. The developer may use semiformal notations like software specification languages to ensure correct refinement of the specifications from functional specification via TOE design down to the implementation level.

In this way the semiformal presentation clearly establishes its accuracy and superiority over informal descriptions.

C.1.2.3 Formal style

A formal specification is expressed within a formal system based upon well-established mathematical concepts. These mathematical concepts are used to define a well-defined semantics, syntax and rules of inference. A formal system is an abstract system of identities and relations that can be described by specifying a formal alphabet, a formal language over that alphabet which is based on a formal syntax, and a set of formal rules of inference for constructing derivations of sentences in the formal language.

The evaluator should examine the identified formal systems to make sure that:

- the semantics, syntax and inference rules of the formal system are defined or a definition is referenced;

- each formal system together with the accompanying explanatory text provides a defined semantics which is characterized by defined meanings of all terms, abbreviations and acronyms that are used in a context other than that accepted by normal usage;

- the formal notation provides rules to determine the meaning of syntactical valid constructs;

- the use of a formal system and a semiformal notation is accompanied by supporting explanatory text in informal style conducive to unambiguous meaning;

- each formal system uses a formal syntax that provides rules to unambiguously recognise constructs;
- each formal system provides proof rules which:
    a) support logical reasoning of well-established mathematical concepts;
    b) help to prevent derivation of contradictions.

If the developer uses a formal system which is already accepted by the evaluation authority, then the evaluator can rely on the level of formality and strength of the system and focus on the instantiation of the formal system to the specifications and correspondence proofs.

The formal style supports mathematical proofs of the security properties based on the formal properties, the consistency of refinements and the correspondence of the representations. Formal tool support is adequate and encouraged whenever manual derivations would otherwise become long winded and incomprehensible. Formal tools are also apt to reduce the error probability inherent in manual derivations.