Australian Government
Department of Defence

Defence Signals Directorate

Australasian Information Security Evaluation Program

Implementation Representation (ADV_IMP.3) - CC V2.2

Common Evaluation Methodology

15 February 2006

Version 1.2

Amendment Record

Version	Date	Description
1.0	29 August 2005	Release
1.1	18 November 2005	Releasable to AISEFs
1.2	15 February 2006	Added CC V2.2 to title

Table of Contents

1 Structured Implementation of the TSF (ADV_IMP.3)

1.1 Objectives

1 The objective of this sub-activity is to determine whether the developer has adopted a structured approach to implementation of the TOE and that the implementation representation is sufficient to satisfy the functional requirements of the ST and is a correct realisation of the low-level design.

1.2 Input

2 The evaluation evidence for this sub-activity is:

a) the ST;

b) the low-level design;

c) the implementation representation.

1.3 Evaluator Actions

1.3.1 ADV_IMP.3.1E

ADV_IMP.3.1C	The implementation representation shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.3-1 The evaluator ***shall examine*** the implementation representation to determine that it unambiguously defines the TSF to a level of detail such that the TSF can be generated without any further design decisions.

3 This work unit requires the evaluator to confirm that the implementation representation is suitable for analysis. The evaluator should consider the process needed to generate the TSF from the representation provided. If the process is well-defined, requiring no further design decisions (for example, requiring only the compilation of source code, or the building of hardware from hardware drawings), then the implementation representation can be said to be suitable.

4 Any programming languages used must be well defined with an unambiguous definition of all statements, as well as the compiler options used to generate the object code. The evaluator may refer to results of the ALC_TAT.1 Well-defined development tools sub-activity when completing this work unit.

ADV_IMP.3-2 The evaluator ***shall examine*** the implementation representation provided by the developer to determine that it is the entire TSF.

5 This work unit can be completed in conjunction with the ADV_RCR.2 Semi-formal correspondence demonstration sub-activity. In this case, the evaluator verifies that the entire TSF as represented in the Low-level design is completely represented in the implementation.

6 A provided implementation representation that does not correspond with the TSF as represented in the Low-level design or that does not match with the TSF for a generated TOE cannot be considered to be the entire TSF.

ADV_IMP.3.2C The implementation representation shall be internally consistent.

ADV_IMP.3-3 The evaluator ***shall examine*** the implementation representation to determine that it is internally consistent.

7 The evaluator looks for inconsistencies by comparing portions of the implementation representation. In the case of source code, for example, if one portion of the source code includes a call to a subprogram in another portion, the evaluator looks to see that the arguments of the calling program match the called program's handling of the arguments. In the case of hardware drawings, the evaluator looks for such things as agreement between the nature and characteristics of the two ends of a circuit trace (e.g. voltage level, direction of logic, signal timing requirements). For guidance on consistency analysis see B.3.

ADV_IMP.3.3C The implementation representation shall describe the relationships between all portions of the implementation.

ADV_IMP.3-4 The evaluator ***shall examine*** the implementation representation to determine that it describes the relationships between all portions of the implementation.

8 The evaluator's objective is to gain an understanding of how different parts of the implementation belong together and to gain assurance of the correctness of the implementation of the TSF. The interrelationships between the single modules might have already been analysed on the LLD representation level (cf. ADV_LLD.1.5C). The relationships within a module represent an object for analysis inherent to the implementation representation level.

9 In the case that the implementation representation includes software source code as well as hardware drawings, understanding the relation between hardware parts and software parts of the implementation representation is of importance.

10 It is of specific importance for interfaces relevant to security that the evaluator understands how different parts of the TSF work together via their interfaces and to understand the behaviour of the TSF in operation.

11 The evaluator can use developer's tools in order to perform the current work unit, e.g. a debugger.

ADV_IMP.3.4C ***The implementation representation shall be structured into small and comprehensible sections.***

ADV_IMP.3-5 The evaluators ***shall examine*** the implementation representation to determine that it is structured into small and comprehensible sections.

12 This determination should be made in conjunction with ADV_INT.1 Modularity sub-activity and the ALC_TAT.1 Well-defined development tools sub-activity. The structure of the implementation representation should be sufficiently detailed for the evaluator to easily comprehend the purpose of each section and how it contributes to the implementation of the TSF.

13 The structuring of the implementation representation into small sections will depend on whether the TOE includes software source code and/or hardware drawings describing the TSF. The evaluator looks to each section of the implementation representation to present a logical grouping of implementation detail.

14 The evaluator should use the developer's implementation standards to comprehend the statements and notations used in the implementation representation.

1.3.2 ADV_IMP.3.2E

ADV_IMP.3.2E The evaluator shall determine that the implementation representation is an accurate and complete instantiation of the TOE security functional requirements.

ADV_IMP.3-6 The evaluator ***shall examine*** the entire implementation representation to determine that it accurately instantiates all TOE security functional requirements.

15 For those portions of the implementation representation that provide security functions directly, the evaluator determines that the implementation matches the TOE security functional requirement. The remaining portions of the implementation representation may support some TOE functional requirement. In making a determination about these remaining portions, the evaluator makes use of the low-level design to assess if the portions in the implementation representation, in combination with other portions as described in the low-level design, work together to instantiate a TOE security functional requirement.

16 The remaining portions of the implementation representation subset, if any, can generally be ignored because they are unrelated to any of the TOE security functional requirements supported by the implementation subset.

17 However, the evaluator should be careful to not overlook any portions that play an indirect role, no matter how distant, in supporting the TOE security functions. For example, in typical operating systems, the source code for portions of the nucleus (or kernel) may not have any direct role in supporting a TOE security function, but are capable of interfering with the correct functioning of those portions of the nucleus that do have a direct role. If any such portions are found to exist in the subset of the implementation representation provided, they should be assessed not to interfere with the portions that do, provided that the ST requires such non-interference. This assessment typically will not require the same level of detailed examination that is required for those portions of the implementation representation that play a more direct role in supporting the TOE security functions.

18 This determination should be made in conjunction with the ADV_RCR.2 Semi-formal correspondence demonstration sub-activity.