Common Criteria for Information Technology Security Evaluation

Catalogue of assurance components

CC:2022

2023-04-20

Table of Contents

[1 Class ACE Protection Profile Configuration evaluation 7](#_Toc132870929)

[1.1 Introduction 7](#_Toc132870930)

[1.2 PP-Module conformance claims (ACE\_CCL) 8](#_Toc132870931)

[1.2.1 PP-Module conformance claims (ACE\_CCL.1) 8](#_Toc132870932)

[1.3 PP-Configuration consistency (ACE\_CCO) 9](#_Toc132870933)

[1.3.1 PP-Configuration consistency (ACE\_CCO.1) 9](#_Toc132870934)

[1.4 PP-Module extended components definition (ACE\_ECD) 12](#_Toc132870935)

[1.4.1 PP-Module extended components definition (ACE\_ECD.1) 12](#_Toc132870936)

[1.5 PP-Module introduction (ACE\_INT) 13](#_Toc132870937)

[1.5.1 PP-Module introduction (ACE\_INT.1) 13](#_Toc132870938)

[1.6 PP-Module consistency (ACE\_MCO) 14](#_Toc132870939)

[1.6.1 PP-Module consistency (ACE\_MCO.1) 14](#_Toc132870940)

[1.7 PP-Module security objectives (ACE\_OBJ) 15](#_Toc132870941)

[1.7.1 PP-Module security objectives for the operational environment (ACE\_OBJ.1) 15](#_Toc132870942)

[1.7.2 PP-Module security objectives (ACE\_OBJ.2) 16](#_Toc132870943)

[1.8 PP-Module security requirements (ACE\_REQ) 16](#_Toc132870944)

[1.8.1 PP-Module stated security requirements (ACE\_REQ.1) 17](#_Toc132870945)

[1.8.2 PP-Module derived security requirements (ACE\_REQ.2) 18](#_Toc132870946)

[1.9 PP-Module security problem definition (ACE\_SPD) 18](#_Toc132870947)

[1.9.1 PP-Module security problem definition (ACE\_SPD.1) 19](#_Toc132870948)

[2 Class ACO Composition 20](#_Toc132870949)

[2.1 Introduction 20](#_Toc132870950)

[2.2 Composition rationale (ACO\_COR) 22](#_Toc132870951)

[2.2.1 Composition rationale (ACO\_COR.1) 22](#_Toc132870952)

[2.3 Composed TOE testing (ACO\_CTT) 23](#_Toc132870953)

[2.3.1 Interface testing (ACO\_CTT.1) 24](#_Toc132870954)

[2.3.2 Rigorous interface testing (ACO\_CTT.2) 25](#_Toc132870955)

[2.4 Development evidence (ACO\_DEV) 25](#_Toc132870956)

[2.4.1 Functional Description (ACO\_DEV.1) 26](#_Toc132870957)

[2.4.2 Basic evidence of design (ACO\_DEV.2) 27](#_Toc132870958)

[2.4.3 Detailed evidence of design (ACO\_DEV.3) 27](#_Toc132870959)

[2.5 Reliance of dependent component (ACO\_REL) 28](#_Toc132870960)

[2.5.1 Basic reliance information (ACO\_REL.1) 29](#_Toc132870961)

[2.5.2 Reliance information (ACO\_REL.2) 29](#_Toc132870962)

[2.6 Composition vulnerability analysis (ACO\_VUL) 30](#_Toc132870963)

[2.6.1 Composition vulnerability review (ACO\_VUL.1) 30](#_Toc132870964)

[2.6.2 Composition vulnerability analysis (ACO\_VUL.2) 31](#_Toc132870965)

[2.6.3 Enhanced-Basic Composition vulnerability analysis (ACO\_VUL.3) 32](#_Toc132870966)

[3 Class ADV Development 33](#_Toc132870967)

[3.1 Introduction 33](#_Toc132870968)

[3.2 Security Architecture (ADV\_ARC) 38](#_Toc132870969)

[3.2.1 Security architecture description (ADV\_ARC.1) 40](#_Toc132870970)

[3.3 Composite design compliance (ADV\_COMP) 40](#_Toc132870971)

[3.3.1 Design compliance with the base component-related user guidance, ETR for composite evaluation and report of the base component evaluation authority (ADV\_COMP.1) 42](#_Toc132870972)

[3.4 Functional specification (ADV\_FSP) 42](#_Toc132870973)

[3.4.1 Basic functional specification (ADV\_FSP.1) 46](#_Toc132870974)

[3.4.2 Security-enforcing functional specification (ADV\_FSP.2) 47](#_Toc132870975)

[3.4.3 Functional specification with complete summary (ADV\_FSP.3) 47](#_Toc132870976)

[3.4.4 Complete functional specification (ADV\_FSP.4) 48](#_Toc132870977)

[3.4.5 Complete semi-formal functional specification with additional error information (ADV\_FSP.5) 49](#_Toc132870978)

[3.4.6 Complete semi-formal functional specification with additional formal specification (ADV\_FSP.6) 49](#_Toc132870979)

[3.5 Implementation representation (ADV\_IMP) 51](#_Toc132870980)

[3.5.1 Implementation representation of the TSF (ADV\_IMP.1) 52](#_Toc132870981)

[3.5.2 Complete mapping of the implementation representation of the TSF (ADV\_IMP.2) 53](#_Toc132870982)

[3.6 TSF internals (ADV\_INT) 53](#_Toc132870983)

[3.6.1 Well-structured subset of TSF internals (ADV\_INT.1) 54](#_Toc132870984)

[3.6.2 Well-structured internals (ADV\_INT.2) 55](#_Toc132870985)

[3.6.3 Minimally complex internals (ADV\_INT.3) 55](#_Toc132870986)

[3.7 Security policy modelling (ADV\_SPM) 56](#_Toc132870987)

[3.7.1 Formal TOE security policy model (ADV\_SPM.1) 57](#_Toc132870988)

[3.8 TOE design (ADV\_TDS) 58](#_Toc132870989)

[3.8.1 Basic design (ADV\_TDS.1) 60](#_Toc132870990)

[3.8.2 Architectural design (ADV\_TDS.2) 61](#_Toc132870991)

[3.8.3 Basic modular design (ADV\_TDS.3) 61](#_Toc132870992)

[3.8.4 Semiformal modular design (ADV\_TDS.4) 62](#_Toc132870993)

[3.8.5 Complete semiformal modular design (ADV\_TDS.5) 63](#_Toc132870994)

[3.8.6 Complete semiformal modular design with formal high-level design presentation (ADV\_TDS.6) 64](#_Toc132870995)

[4 Class AGD Guidance documents 66](#_Toc132870996)

[4.1 Introduction 66](#_Toc132870997)

[4.2 Operational user guidance (AGD\_OPE) 66](#_Toc132870998)

[4.2.1 Operational user guidance (AGD\_OPE.1) 68](#_Toc132870999)

[4.3 Preparative procedures (AGD\_PRE) 68](#_Toc132871000)

[4.3.1 Preparative procedures (AGD\_PRE.1) 69](#_Toc132871001)

[5 Class ALC Life-cycle support 70](#_Toc132871002)

[5.1 Introduction 70](#_Toc132871003)

[5.2 CM capabilities (ALC\_CMC) 71](#_Toc132871004)

[5.2.1 Labelling of the TOE (ALC\_CMC.1) 73](#_Toc132871005)

[5.2.2 Use of the CM system (ALC\_CMC.2) 73](#_Toc132871006)

[5.2.3 Authorization controls (ALC\_CMC.3) 74](#_Toc132871007)

[5.2.4 Production support, acceptance procedures and automation (ALC\_CMC.4) 74](#_Toc132871008)

[5.2.5 Advanced support (ALC\_CMC.5) 75](#_Toc132871009)

[5.3 CM scope (ALC\_CMS) 76](#_Toc132871010)

[5.3.1 TOE CM coverage (ALC\_CMS.1) 77](#_Toc132871011)

[5.3.2 Parts of the TOE CM coverage (ALC\_CMS.2) 77](#_Toc132871012)

[5.3.3 Implementation representation CM coverage (ALC\_CMS.3) 78](#_Toc132871013)

[5.3.4 Problem tracking CM coverage (ALC\_CMS.4) 78](#_Toc132871014)

[5.3.5 Development tools CM coverage (ALC\_CMS.5) 78](#_Toc132871015)

[5.4 Integration of composition parts and consistency check of delivery procedures (ALC\_COMP) 79](#_Toc132871016)

[5.4.1 Integration of the dependent component into the related base component and Consistency check for delivery and acceptance procedures (ALC\_COMP.1) 80](#_Toc132871017)

[5.5 Delivery (ALC\_DEL) 80](#_Toc132871018)

[5.5.1 Delivery procedures (ALC\_DEL.1) 81](#_Toc132871019)

[5.6 Developer environment security (ALC\_DVS) 82](#_Toc132871020)

[5.6.1 Identification of security controls (ALC\_DVS.1) 82](#_Toc132871021)

[5.6.2 Sufficiency of security controls (ALC\_DVS.2) 83](#_Toc132871022)

[5.7 Flaw remediation (ALC\_FLR) 83](#_Toc132871023)

[5.7.1 Basic flaw remediation (ALC\_FLR.1) 84](#_Toc132871024)

[5.7.2 Flaw reporting procedures (ALC\_FLR.2) 85](#_Toc132871025)

[5.7.3 Systematic flaw remediation (ALC\_FLR.3) 86](#_Toc132871026)

[5.8 Development Life-cycle definition (ALC\_LCD) 87](#_Toc132871027)

[5.8.1 Developer defined life-cycle processes (ALC\_LCD.1) 88](#_Toc132871028)

[5.8.2 Measurable life-cycle model (ALC\_LCD.2) 88](#_Toc132871029)

[5.9 Tools and techniques (ALC\_TAT) 89](#_Toc132871030)

[5.9.1 Well-defined development tools (ALC\_TAT.1) 90](#_Toc132871031)

[5.9.2 Compliance with implementation standards (ALC\_TAT.2) 90](#_Toc132871032)

[5.9.3 Compliance with implementation standards - all parts (ALC\_TAT.3) 91](#_Toc132871033)

[5.10 TOE Development Artefacts (ALC\_TDA) 92](#_Toc132871034)

[5.10.1 Uniquely identifying implementation representation (ALC\_TDA.1) 93](#_Toc132871035)

[5.10.2 Matching CMS scope of implementation representation (ALC\_TDA.2) 94](#_Toc132871036)

[5.10.3 Regenerate TOE with well-defined development tools (ALC\_TDA.3) 96](#_Toc132871037)

[6 Class APE Protection Profile (PP) evaluation 99](#_Toc132871038)

[6.1 Introduction 99](#_Toc132871039)

[6.2 Conformance claims (APE\_CCL) 99](#_Toc132871040)

[6.2.1 Conformance claims (APE\_CCL.1) 100](#_Toc132871041)

[6.3 Extended components definition (APE\_ECD) 101](#_Toc132871042)

[6.3.1 Extended components definition (APE\_ECD.1) 101](#_Toc132871043)

[6.4 PP introduction (APE\_INT) 102](#_Toc132871044)

[6.4.1 PP introduction (APE\_INT.1) 102](#_Toc132871045)

[6.5 Security objectives (APE\_OBJ) 103](#_Toc132871046)

[6.5.1 Security objectives for the operational environment (APE\_OBJ.1) 103](#_Toc132871047)

[6.5.2 Security objectives (APE\_OBJ.2) 104](#_Toc132871048)

[6.6 Security requirements (APE\_REQ) 104](#_Toc132871049)

[6.6.1 Direct rationale PP-Module security requirements (APE\_REQ.1) 105](#_Toc132871050)

[6.6.2 Derived security requirements (APE\_REQ.2) 105](#_Toc132871051)

[6.7 Security problem definition (APE\_SPD) 106](#_Toc132871052)

[6.7.1 Security problem definition (APE\_SPD.1) 106](#_Toc132871053)

[7 Class ASE Security Target (ST) evaluation 108](#_Toc132871054)

[7.1 Introduction 108](#_Toc132871055)

[7.2 Conformance claims (ASE\_CCL) 109](#_Toc132871056)

[7.2.1 Conformance claims (ASE\_CCL.1) 109](#_Toc132871057)

[7.3 Consistency of composite product Security Target (ASE\_COMP) 110](#_Toc132871058)

[7.3.1 Consistency of Security Target (ST) (ASE\_COMP.1) 112](#_Toc132871059)

[7.4 Extended components definition (ASE\_ECD) 112](#_Toc132871060)

[7.4.1 Extended components definition (ASE\_ECD.1) 113](#_Toc132871061)

[7.5 ST introduction (ASE\_INT) 113](#_Toc132871062)

[7.5.1 ST introduction (ASE\_INT.1) 113](#_Toc132871063)

[7.6 Security objectives (ASE\_OBJ) 114](#_Toc132871064)

[7.6.1 Security objectives for the operational environment (ASE\_OBJ.1) 115](#_Toc132871065)

[7.6.2 Security objectives (ASE\_OBJ.2) 115](#_Toc132871066)

[7.7 Security requirements (ASE\_REQ) 116](#_Toc132871067)

[7.7.1 Direct rationale security requirements (ASE\_REQ.1) 116](#_Toc132871068)

[7.7.2 Derived security requirements (ASE\_REQ.2) 117](#_Toc132871069)

[7.8 Security problem definition (ASE\_SPD) 118](#_Toc132871070)

[7.8.1 Security problem definition (ASE\_SPD.1) 118](#_Toc132871071)

[7.9 TOE summary specification (ASE\_TSS) 119](#_Toc132871072)

[7.9.1 TOE summary specification (ASE\_TSS.1) 119](#_Toc132871073)

[7.9.2 TOE summary specification with architectural design summary (ASE\_TSS.2) 120](#_Toc132871074)

[8 Class ATE Tests 121](#_Toc132871075)

[8.1 Introduction 121](#_Toc132871076)

[8.2 Composite functional testing (ATE\_COMP) 122](#_Toc132871077)

[8.2.1 Composite product functional testing (ATE\_COMP.1) 123](#_Toc132871078)

[8.3 Coverage (ATE\_COV) 123](#_Toc132871079)

[8.3.1 Evidence of coverage (ATE\_COV.1) 124](#_Toc132871080)

[8.3.2 Analysis of coverage (ATE\_COV.2) 124](#_Toc132871081)

[8.3.3 Rigorous analysis of coverage (ATE\_COV.3) 124](#_Toc132871082)

[8.4 Depth (ATE\_DPT) 125](#_Toc132871083)

[8.4.1 Testing: basic design (ATE\_DPT.1) 125](#_Toc132871084)

[8.4.2 Testing: security enforcing modules (ATE\_DPT.2) 126](#_Toc132871085)

[8.4.3 Testing: modular design (ATE\_DPT.3) 126](#_Toc132871086)

[8.4.4 Testing: implementation representation (ATE\_DPT.4) 127](#_Toc132871087)

[8.5 Functional tests (ATE\_FUN) 127](#_Toc132871088)

[8.5.1 Functional testing (ATE\_FUN.1) 128](#_Toc132871089)

[8.5.2 Ordered functional testing (ATE\_FUN.2) 129](#_Toc132871090)

[8.6 Independent testing (ATE\_IND) 129](#_Toc132871091)

[8.6.1 Independent testing - conformance (ATE\_IND.1) 130](#_Toc132871092)

[8.6.2 Independent testing - sample (ATE\_IND.2) 131](#_Toc132871093)

[8.6.3 Independent testing - complete (ATE\_IND.3) 131](#_Toc132871094)

[9 Class AVA Vulnerability assessment 132](#_Toc132871095)

[9.1 Introduction 132](#_Toc132871096)

[9.2 Application notes 132](#_Toc132871097)

[9.3 Composite vulnerability assessment (AVA\_COMP) 133](#_Toc132871098)

[9.3.1 Composite product vulnerability assessment (AVA\_COMP.1) 134](#_Toc132871099)

[9.4 Vulnerability analysis (AVA\_VAN) 134](#_Toc132871100)

[9.4.1 Vulnerability survey (AVA\_VAN.1) 135](#_Toc132871101)

[9.4.2 Vulnerability analysis (AVA\_VAN.2) 135](#_Toc132871102)

[9.4.3 Focused vulnerability analysis (AVA\_VAN.3) 136](#_Toc132871103)

[9.4.4 Methodical vulnerability analysis (AVA\_VAN.4) 137](#_Toc132871104)

[9.4.5 Advanced methodical vulnerability analysis (AVA\_VAN.5) 137](#_Toc132871105)

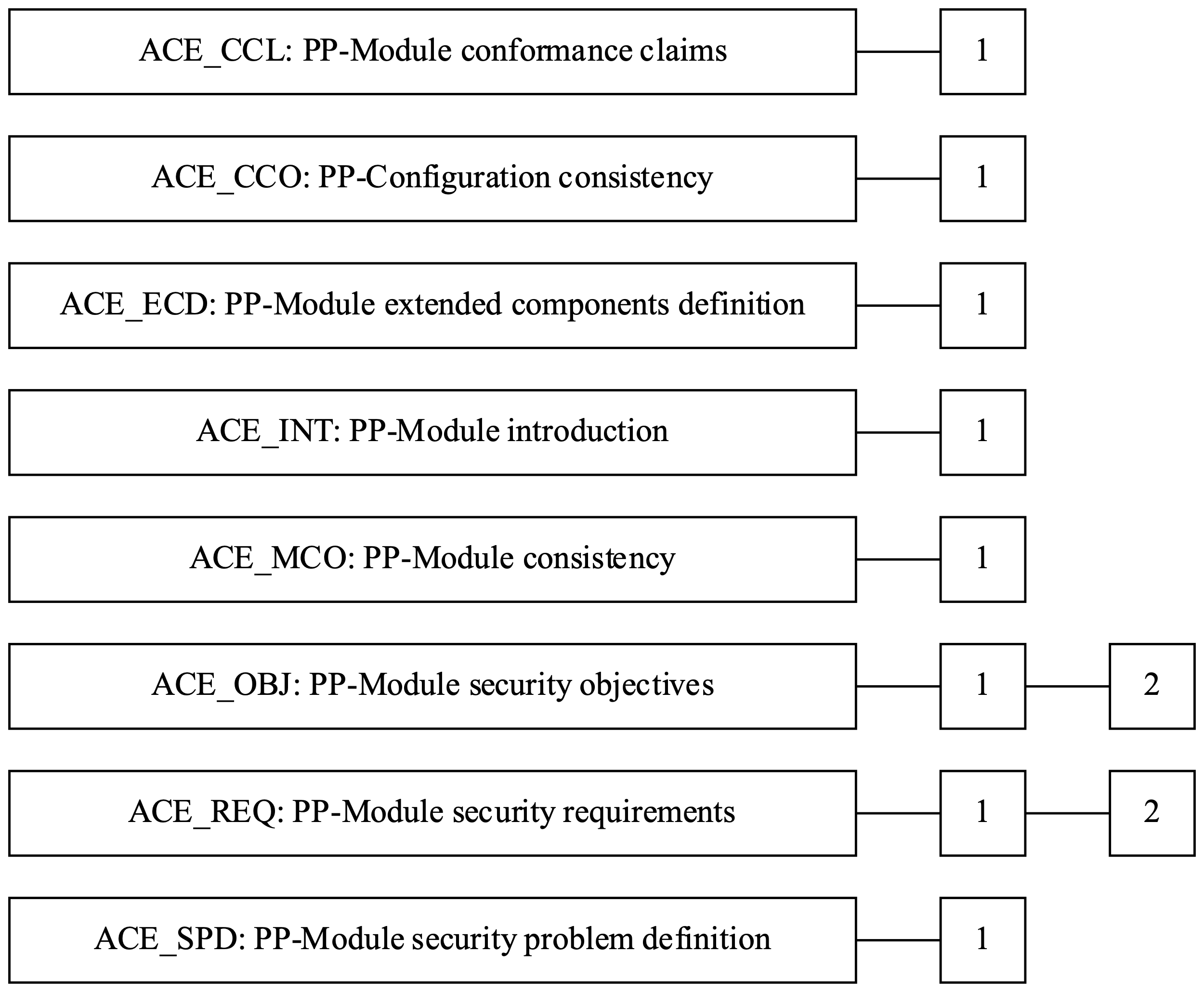
# Class ACE Protection Profile Configuration evaluation

## Introduction

Evaluating a PP-Configuration is required to demonstrate that the PP-Configuration is sound and consistent. These properties are necessary for the PP-Configuration to be suitable for use as the basis for writing an ST.

1. The class [ACE](#ace) is defined for the evaluation of a PP-Configuration composed of at least one PP and one other component (PPs and/or PP-Modules). The evaluation of PPs is addressed in Class [APE](#ape). The class [ACE](#ace) defines the requirements for:

* Evaluating the PP-Modules in the framework of their PP-Modules Base(s) (components [ACE\_INT.1](#ace_int.1), [ACE\_CCL.1](#ace_ccl.1), [ACE\_SPD.1](#ace_spd.1), [ACE\_OBJ.1](#ace_obj.1) or [ACE\_OBJ.2](#ace_obj.2), [ACE\_REQ.1](#ace_req.1) or [ACE\_REQ.2](#ace_req.2), and [ACE\_MCO.1](#ace_mco.1)).
* Evaluating the consistency of the combination of all the PPs and PP-Modules that belong to the PP-Configuration (see [ACE\_CCO.1](#ace_cco.1)).



|  | [ACE\_CCL.1](#ace_ccl.1) | [ACE\_ECD.1](#ace_ecd.1) | [ACE\_INT.1](#ace_int.1) | [ACE\_MCO.1](#ace_mco.1) | [ACE\_OBJ.1](#ace_obj.1) | [ACE\_OBJ.2](#ace_obj.2) | [ACE\_REQ.1](#ace_req.1) | [ACE\_REQ.2](#ace_req.2) | [ACE\_SPD.1](#ace_spd.1) | [APE\_ECD.1](#ape_ecd.1) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**ACE\_CCL.1**](#ace_ccl.1) |  | X | X |  |  | - | O1 | O1 | - | - |
| [**ACE\_CCO.1**](#ace_cco.1) | X | X | X | X | O1 | O1 | O2 | O2 | X | - |
| [**ACE\_ECD.1**](#ace_ecd.1) |  |  |  |  |  |  |  |  |  |  |
| [**ACE\_INT.1**](#ace_int.1) |  |  |  |  |  |  |  |  |  |  |
| [**ACE\_MCO.1**](#ace_mco.1) |  | - | X |  | O1 | O1 | O2 | O2 | X | - |
| [**ACE\_OBJ.1**](#ace_obj.1) |  |  |  |  |  |  |  |  |  |  |
| [**ACE\_OBJ.2**](#ace_obj.2) |  |  |  |  |  |  |  |  | X |  |
| [**ACE\_REQ.1**](#ace_req.1) |  |  |  |  |  |  |  |  | X | X |
| [**ACE\_REQ.2**](#ace_req.2) |  | X |  |  |  | X |  |  | - |  |
| [**ACE\_SPD.1**](#ace_spd.1) |  |  |  |  |  |  |  |  |  |  |

## PP-Module conformance claims (ACE\_CCL)

Objectives

1. The objective of this family is to determine the validity of the conformance claim and conformance statement. A PP-Module cannot claim conformance to any PP, PP-Configuration, or another PP-Module.

### PP-Module conformance claims (ACE\_CCL.1)

Dependencies: [ACE\_INT.1](#ace_int.1), [ACE\_ECD.1](#ace_ecd.1), [[ACE\_REQ.1](#ace_req.1) or [ACE\_REQ.2](#ace_req.2)]

Developer action elements

ACE\_CCL.1.1D The developer shall provide a conformance claim.

ACE\_CCL.1.2D The developer shall provide a conformance statement.

Content and presentation elements

ACE\_CCL.1.1C The conformance claim shall identify the ISO/IEC 15408 edition to which the PP-Module claims conformance.

ACE\_CCL.1.2C The conformance claim shall describe the conformance of the PP-Module to ISO/IEC 15408-2 as either ISO/IEC 15408-2 conformant or ISO/IEC 15408-2 extended.

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.

ACE\_CCL.1.4C The conformance claim shall describe the conformance of the PP-Module to this document as either “ISO/IEC 15408-3 conformant” or “ISO/IEC 15408-3 extended”.

ACE\_CCL.1.5C The conformance claim shall be consistent with the extended components definition.

ACE\_CCL.1.6C The conformance claim shall identify all functional packages to which the PP-Module claims conformance.

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.

ACE\_CCL.1.8C The conformance claim shall identify all assurance packages to which the PP-Module claims conformance.

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.

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.

ACE\_CCL.1.11C The conformance statement may identify the set of ISO/IEC 18045-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.

Evaluator action elements

ACE\_CCL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## PP-Configuration consistency (ACE\_CCO)

Objectives

1. The objective of this family is to determine the well-formedness and the consistency of the PP-Configuration.

### PP-Configuration consistency (ACE\_CCO.1)

Dependencies: [ACE\_INT.1](#ace_int.1), [ACE\_CCL.1](#ace_ccl.1), [ACE\_SPD.1](#ace_spd.1), [[ACE\_OBJ.1](#ace_obj.1) or [ACE\_OBJ.2](#ace_obj.2)], [ACE\_ECD.1](#ace_ecd.1), [[ACE\_REQ.1](#ace_req.1) or [ACE\_REQ.2](#ace_req.2)], [ACE\_MCO.1](#ace_mco.1)

Developer action elements

ACE\_CCO.1.1D The developer shall provide the reference of the PP-Configuration.

ACE\_CCO.1.2D The developer shall provide a components statement.

ACE\_CCO.1.3D The developer shall provide a TOE overview.

ACE\_CCO.1.4D The developer shall provide a conformance claim.

ACE\_CCO.1.5D The developer shall provide a conformance statement within the conformance claim.

ACE\_CCO.1.6D The developer shall provide a consistency rationale.

ACE\_CCO.1.7D The developer shall provide a SAR statement.

ACE\_CCO.1.8D The developer shall provide the set of evaluation methods and/or activities that are applicable to the PP-Configuration.

Content and presentation elements

ACE\_CCO.1.1C The PP-Configuration reference shall uniquely identify the PP-Configuration.

ACE\_CCO.1.2C The PP-Configuration components statement shall uniquely identify the PPs and PP-Modules that compose the PP-Configuration.

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.

ACE\_CCO.1.4C For a multi-assurance PP-Configuration, the components statement shall describe the organization of the TSF in terms of the sub-TSFs defined in the PPs and PP-Modules defined in the PP-Configuration.

ACE\_CCO.1.5C The TOE overview shall identify the TOE type.

ACE\_CCO.1.6C The TOE overview shall describe the usage and major security features of the TOE.

ACE\_CCO.1.7C The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.

ACE\_CCO.1.8C The conformance claim shall identify the ISO/IEC 15408 edition(s) to which the PP-Configuration components claim conformance.

ACE\_CCO.1.9C The conformance claim shall describe the conformance of the PP-Configuration to ISO/IEC 15408-2 as either ISO/IEC 15408-2 conformant or ISO/IEC 15408-2 extended.

ACE\_CCO.1.10C The conformance claim shall describe the conformance of the PP-Configuration to this document as either “ISO/IEC 15408-3 conformant” or ISO/IEC 15408-3 extended.”

ACE\_CCO.1.11C The conformance claim shall be consistent with the conformance claims of the PP-Configuration components.

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.

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.

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.

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.

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 ISO/IEC 18045-derived Evaluation methods and Evaluation activities that are applicable to the PP-Configuration under evaluation.

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.

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.

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.

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.

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:

1. that the global set of SARs is consistent with the threats as defined in the SPDs of the PP-Configuration components, and
2. that the global set of SARs and the sets of SARs for each sub-TSF are consistent with each other.

Evaluator action elements

ACE\_CCO.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACE\_CCO.1.2E The evaluator shall check that the PP-Configuration consisting of all the PPs and PP-Modules identified in the component statement is consistent.

## PP-Module extended components definition (ACE\_ECD)

Objectives

1. Extended SFRs are requirements that are not based on components from ISO/IEC 15408-2 or this document, but which are based on extended components: components defined by the PP-Module author.
2. Evaluation of the definition of extended functional components is necessary to determine that they are clear and unambiguous, and that they are necessary, i.e. they may not be clearly expressed using existing ISO/IEC 15408-2 or this document components.

### PP-Module extended components definition (ACE\_ECD.1)

Developer action elements

ACE\_ECD.1.1D The developer shall provide a statement of security requirements for the PP-Module.

ACE\_ECD.1.2D The developer shall provide an extended components definition for the PP-Module.

Content and presentation elements

ACE\_ECD.1.1C The statement of security requirements shall identify all the extended security requirements.

ACE\_ECD.1.2C The extended components definition shall define an extended component for each extended security requirement.

ACE\_ECD.1.3C The extended components definition shall describe how each extended component is related to the existing ISO/IEC 15408 series components, families, and classes.

ACE\_ECD.1.4C The extended components definition shall use the existing ISO/IEC 15408 series components, families, classes, and methodology as a model for presentation.

ACE\_ECD.1.5C The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements may be demonstrated

Evaluator action elements

ACE\_ECD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACE\_ECD.1.2E The evaluator shall confirm that no extended component may be clearly expressed using existing components.

## PP-Module introduction (ACE\_INT)

Objectives

1. The objective of this family is to describe the TOE in a narrative way.
2. The evaluation of the PP-Module introduction is required to demonstrate that the PP-Module is correctly identified, and that the PP-Module reference and TOE overview are consistent with each other.

### PP-Module introduction (ACE\_INT.1)

Developer action elements

ACE\_INT.1.1D The developer shall provide a PP-Module introduction.

Content and presentation elements

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.

ACE\_INT.1.2C The PP-Module reference shall uniquely identify the PP-Module.

ACE\_INT.1.3C The identification of the PP-Module Base shall consist of a list of at least one PP and possibly other PPs and PP-Modules on which the PP-Module depends.

ACE\_INT.1.4C The identification of the PP-Module Base(s) shall describe the dependency structure of the PP-Module Base(s).

ACE\_INT.1.5C The PP-Module introduction shall contain as many TOE overviews as alternative PP-Module Bases.

ACE\_INT.1.6C The TOE overview shall summarize the usage and major security features of the TOE.

ACE\_INT.1.7C The TOE overview shall identify the TOE type.

ACE\_INT.1.8C The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.

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).

Evaluator action elements

ACE\_INT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## PP-Module consistency (ACE\_MCO)

Objectives

1. The objective of this family is to determine the consistency of the PP-Module and to state the correspondence between the PP-Module and its PP-Module Base(s).

### PP-Module consistency (ACE\_MCO.1)

Dependencies: [ACE\_INT.1](#ace_int.1), [ACE\_SPD.1](#ace_spd.1), [[ACE\_OBJ.1](#ace_obj.1) or [ACE\_OBJ.2](#ace_obj.2)], [[ACE\_REQ.1](#ace_req.1) or [ACE\_REQ.2](#ace_req.2)]

Developer action elements

ACE\_MCO.1.1D The developer shall provide a consistency rationale of the PP-Module for each of the alternative PP-Module Bases identified in the PP-Module introduction.

ACE\_MCO.1.2D The developer shall provide an assurance rationale of the PP-Module for each of the alternative PP-Module Bases identified in the PP-Module introduction.

Content and presentation elements

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.

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.

ACE\_MCO.1.3C The consistency rationale shall demonstrate that:

1. the statement of the security problem definition is consistent with the statement of the security problem definition of its PP-Module Base(s);
2. 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.

ACE\_MCO.1.4C The consistency rationale shall demonstrate that:

1. the security objectives definition is consistent with the security objectives of its PP-Module Base(s);
2. the security objectives definition is consistent with the security objectives of any functional package for which conformance is being claimed.

ACE\_MCO.1.5C The consistency rationale shall demonstrate that:

1. the security functional requirements definition is consistent with the security functional requirements of its PP-Modules Base(s);
2. the security functional requirements definition is consistent with the security functional requirements of any functional package for which conformance is being claimed.

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.

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).

Evaluator action elements

ACE\_MCO.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. If the PP-Module specifies alternative PP-Module Bases, the evaluator shall perform this action for each consistency rationale.

## PP-Module security objectives (ACE\_OBJ)

Objectives

1. The security objectives are a concise statement of the intended response to the security problem defined through the [APE\_SPD](#ape_spd) family.
2. Evaluation of the security objectives is required to demonstrate that 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.

Levelling criteria

1. The components in this family are levelled on whether they prescribe only security objectives for the operational environment (see [ACE\_OBJ.1](#ace_obj.1)), or also security objectives for the TOE (see [ACE\_OBJ.2](#ace_obj.2)).

### PP-Module security objectives for the operational environment (ACE\_OBJ.1)

Developer action elements

ACE\_OBJ.1.1D The developer shall provide a statement of security objectives for the operational environment of the PP-Module.

ACE\_OBJ.1.2D The developer shall provide a security objectives rationale for the operational environment of the PP-Module.

Content and presentation elements

ACE\_OBJ.1.1C The statement of security objectives shall describe the security objectives for the operational environment.

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.

ACE\_OBJ.1.3C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

ACE\_OBJ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### PP-Module security objectives (ACE\_OBJ.2)

Dependencies: [ACE\_SPD.1](#ace_spd.1)

Developer action elements

ACE\_OBJ.2.1D The developer shall provide a statement of security objectives for the PP-Module.

ACE\_OBJ.2.2D The developer shall provide a security objectives rationale for the PP-Module.

Content and presentation elements

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.

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.

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.

ACE\_OBJ.2.4C The security objectives **rationale** shall **demonstrate** **that** the security objectives **counter** **all** **threats.**

ACE\_OBJ.2.5C The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.

ACE\_OBJ.2.6C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

ACE\_OBJ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## PP-Module security requirements (ACE\_REQ)

Objectives

1. The SFRs form a clear, unambiguous and well-defined description of the expected security behaviour of the TOE. The SARs form a clear, unambiguous and well-defined description of the expected activities that will be undertaken to gain assurance in the TOE.
2. Evaluation of the security requirements is required to ensure that they are clear, unambiguous and well-defined.

Levelling criteria

1. The components in this family are levelled on whether the SFRs are derived from SPD (see [ACE\_REQ.1](#ace_req.1)), or whether the SFRs are derived from the security objectives for the TOE (see [ACE\_REQ.2](#ace_req.2).).

### PP-Module stated security requirements (ACE\_REQ.1)

Dependencies: [APE\_ECD.1](#ape_ecd.1), [ACE\_SPD.1](#ace_spd.1)

Developer action elements

ACE\_REQ.1.1D The developer shall provide a statement of security requirements for the PP-Module.

ACE\_REQ.1.2D The developer shall provide a security requirements rationale for the PP-Module.

Content and presentation elements

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)).

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.

ACE\_REQ.1.3C The statement of security requirements shall identify all operations on the security requirements.

ACE\_REQ.1.4C All operations shall be performed correctly.

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.

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.

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.

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.

ACE\_REQ.1.9C The security requirements rationale shall explain why the SARs were chosen.

ACE\_REQ.1.10C The statement of security requirements shall be internally consistent.

Evaluator action elements

ACE\_REQ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### PP-Module derived security requirements (ACE\_REQ.2)

Dependencies: [ACE\_ECD.1](#ace_ecd.1), [ACE\_OBJ.2](#ace_obj.2)

Developer action elements

ACE\_REQ.2.1D The developer shall provide a statement of security requirements for the PP-Module.

ACE\_REQ.2.2D The developer shall provide a security requirements rationale for the PP-Module.

Content and presentation elements

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)).

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.

ACE\_REQ.2.3C The statement of security requirements shall identify all operations on the security requirements.

ACE\_REQ.2.4C All operations shall be performed correctly.

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.

ACE\_REQ.2.6C The security requirements rationale shall trace each SFR back to the **security** **objectives** **for** the **TOE** enforced by that SFR.

ACE\_REQ.2.7C The security requirements rationale shall demonstrate that the SFRs **meet** **all** security objectives for the TOE.

ACE\_REQ.2.8C The security requirements rationale shall explain why the SARs were chosen.

ACE\_REQ.2.9C The statement of security requirements shall be internally consistent.

Evaluator action elements

ACE\_REQ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## PP-Module security problem definition (ACE\_SPD)

Objectives

1. This part of the PP-Module defines the security problem to be addressed by the TOE and the operational environment of the TOE.
2. Evaluation of the security problem definition is required to demonstrate that the security problem intended to be addressed by the TOE and its operational environment, is clearly defined.

### PP-Module security problem definition (ACE\_SPD.1)

Developer action elements

ACE\_SPD.1.1D The developer shall provide a security problem definition.

Content and presentation elements

ACE\_SPD.1.1C The security problem definition shall describe the threats.

ACE\_SPD.1.2C All threats shall be described in terms of a threat agent, an asset, and an adverse action.

ACE\_SPD.1.3C The security problem definition shall describe the OSPs.

ACE\_SPD.1.4C The security problem definition shall describe the assumptions about the operational environment of the TOE.

Evaluator action elements

ACE\_SPD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

# Class ACO Composition

## Introduction

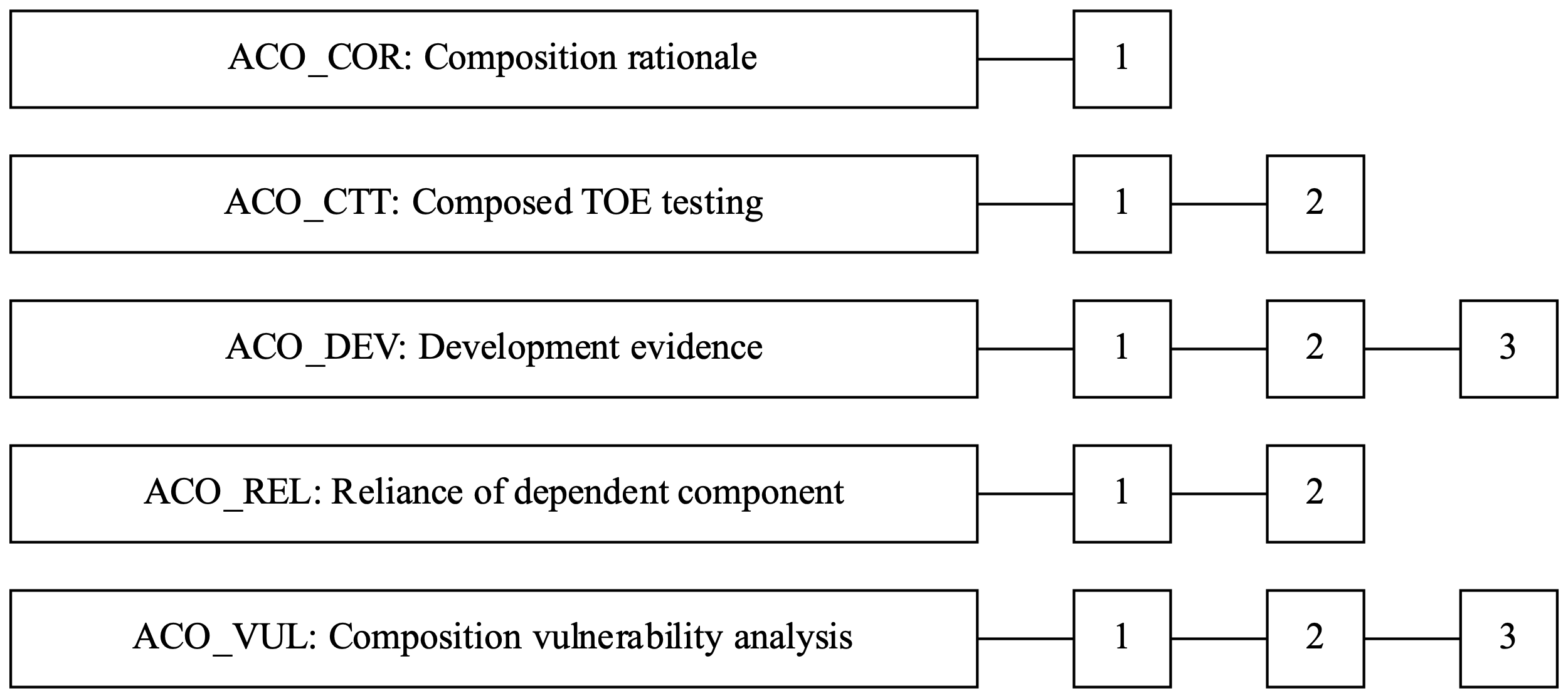
The class [ACO](#aco) encompasses five families. These families specify assurance requirements that are designed to provide confidence that a composed TOE will operate securely when relying upon security functionality provided by previously evaluated software, firmware or hardware components.

1. Composition involves taking two or more IT entities successfully evaluated against the ISO/IEC 15408 series security assurance requirements packages (base components and dependent components, see Annex B) and combining them for use, with no further development of either IT entity. The development of additional IT entities is not included (entities that have not previously been the subject of a component evaluation). The composed TOE forms a new product that can be installed and integrated into any specific environment instance that meets the objectives for the environment.
2. This approach does not provide an alternative approach for the evaluation of components. under [ACO](#aco) provides a composed TOE integrator a method, which can be used as an alternative to other assurance levels specified in the ISO/IEC 15408 series, to gain confidence in a TOE that is the combination of two or more successfully evaluated components without having to re-evaluate the composite TSF. The composed TOE integrator is referred to as “developer” throughout the [ACO](#aco) class, with any references to the developer of the base or dependent components clarified as such.
3. CAPs, as defined in ISO/IEC 15408-5 provide an assurance scale for composed TOEs. This assurance scale is required in addition to other assurance packages, for example the EALs, because to combine components evaluated against another assurance package and gain equivalent assurance in the resulting composed TOE, all SARs shall be applied to the composed TOE. Although reuse can be made of the component TOE evaluation results, there are often additional aspects of the components that have to be considered in the composed TOE, as described in B.3. Due to the different parties involved in a composed TOE evaluation activity it is generally not possible to gain all necessary evidence about these additional aspects of the components to apply the appropriate EAL. Hence, CAPs have been defined to address the issue of combining evaluated components and gaining a meaningful result. This is discussed further in Annex B.

Key

| A | dependent component-a |
| --- | --- |
| B | base component-b |
| 1 | [ACO\_REL](#aco_rel) (component-a) |
| 2 | [ADV\_FSP](#adv_fsp) (component-b) |
| 3 | [ACO\_DEV](#aco_dev) (component-b) |

1. In a composed TOE it is generally the case that one component relies on the services provided by another component. The component requiring services is termed the dependent component and the component providing the services is termed the base component. This interaction and distinct is discussed further in Annex B. It is assumed to be the case that the developer of the dependent component is supporting the composed TOE evaluation in some manner (as developer, sponsor, or just cooperating and providing the necessary evaluation evidence from the dependent component evaluation) The [ACO](#aco) components included in the CAP assurance packages should not be used as augmentations for component TOE evaluations, as this would provide no meaningful assurance for the component.
2. The families within the [ACO](#aco) class interact in a similar manner to the [ADV](#adv), [ATE](#ate) and [AVA](#ava) classes in a component TOE evaluation and hence leverage from the specification of requirements from those classes where applicable. There are however a few items specific to composed TOE evaluations. To determine how the components interact and identify any deviations from the evaluations of the components, the dependencies that the dependent component has upon the underlying base component are identified ([ACO\_REL](#aco_rel)). This reliance on the base component is specified in terms of the interfaces through which the dependent component makes calls for services in support of the dependent component SFRs. The interfaces, and at higher levels the supporting behaviour, provided by the base component in response to those service requests are analysed in [ACO\_DEV](#aco_dev). The [ACO\_DEV](#aco_dev) family is based on the [ADV\_TDS](#adv_tds) family, as at the simplest level the TSF of each component can be viewed as a subsystem of the composed TOE, with additional portions of each component seen as additional subsystems. Therefore, the interfaces between the components are seen as interactions between subsystems in a component TOE evaluation.
3. It is possible that the interfaces and supporting behaviour descriptions provided for [ACO\_DEV](#aco_dev) are incomplete. This is determined during the conduct of [ACO\_COR](#aco_cor). The [ACO\_COR](#aco_cor) family takes the outputs of [ACO\_REL](#aco_rel) and [ACO\_DEV](#aco_dev) and determines whether the components are being used in their evaluated configuration and identifies where any specifications are incomplete, which are then identified as inputs into testing ([ACO\_CTT](#aco_ctt)) and vulnerability analysis ([ACO\_VUL](#aco_vul)) activities of the composed TOE.
4. Testing of the composed TOE is performed to determine that the composed TOE exhibits the expected behaviour as determined by the composed TOE SFRs, and at higher levels demonstrates the compatibility of the interfaces between the components of the composed TOE.
5. The vulnerability analysis of the composed TOE leverages from the outputs of the vulnerability analysis of the component evaluations. The composed TOE vulnerability analysis considers any residual vulnerabilities from the component evaluations to determine that the residual vulnerabilities are not applicable to the composed TOE. A search of publicly available information relating to the components is also performed to identify any issues reported in the components since the completion of the respective evaluations.
6. The interaction between the [ACO](#aco) families is depicted in Figure 14 below. This shows by solid arrowed lines where the evidence and understanding gained in one family feeds into the next activity and the dashed arrows identify where an activity explicitly traces back to the composed TOE SFRs, as described above.
7. Further discussion of the definition and interactions within composed TOEs is provided in Annex B.



|  | [ACO\_DEV.1](#aco_dev.1) | [ACO\_DEV.2](#aco_dev.2) | [ACO\_DEV.3](#aco_dev.3) | [ACO\_REL.1](#aco_rel.1) | [ACO\_REL.2](#aco_rel.2) | [ALC\_CMC.1](#alc_cmc.1) |
| --- | --- | --- | --- | --- | --- | --- |
| [**ACO\_COR.1**](#aco_cor.1) | X |  |  | X |  | X |
| [**ACO\_CTT.1**](#aco_ctt.1) | X |  |  | X |  |  |
| [**ACO\_CTT.2**](#aco_ctt.2) |  | X |  | - | X |  |
| [**ACO\_DEV.1**](#aco_dev.1) |  |  |  | X |  |  |
| [**ACO\_DEV.2**](#aco_dev.2) |  |  |  | X |  |  |
| [**ACO\_DEV.3**](#aco_dev.3) |  |  |  |  | X |  |
| [**ACO\_REL.1**](#aco_rel.1) |  |  |  |  |  |  |
| [**ACO\_REL.2**](#aco_rel.2) |  |  |  |  |  |  |
| [**ACO\_VUL.1**](#aco_vul.1) | X |  |  | - |  |  |
| [**ACO\_VUL.2**](#aco_vul.2) |  | X |  | - |  |  |
| [**ACO\_VUL.3**](#aco_vul.3) |  |  | X |  | - |  |

## Composition rationale (ACO\_COR)

Objectives

1. This family addresses the requirement to demonstrate that the base component can provide an appropriate level of assurance for use in composition.

Levelling criteria

1. There is only a single component in this family.

### Composition rationale (ACO\_COR.1)

Dependencies: [ACO\_DEV.1](#aco_dev.1), [ALC\_CMC.1](#alc_cmc.1), [ACO\_REL.1](#aco_rel.1)

Developer action elements

ACO\_COR.1.1D The developer shall provide composition rationale for the base component.

Content and presentation elements

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.

Evaluator action elements

ACO\_COR.1.1E The evaluator shall confirm that the information meets all requirements for content and presentation of evidence.

## Composed TOE testing (ACO\_CTT)

Objectives

1. This family requires that testing of composed TOE and testing of the base component, as used in the composed TOE, is performed.

Levelling criteria

1. The components in this family are levelled on the basis of increasing rigour of interface testing and increasing rigour of the analysis of the sufficiency of the tests to demonstrate that the composed TSF operates in accordance with the reliance information and the composed TOE SFRs.

Application notes

1. There are two distinct aspects of testing associated with this family:
2. testing of the interfaces between the base component and the dependent component, which the dependent component rely upon for enforcement of security functionality, to demonstrate their compatibility;
3. testing of the composed TOE to demonstrate that the TOE behaves in accordance with the SFRs for the composed TOE.

If the test configurations used during evaluation of the dependent component included use of the base component as a “platform” and the test analysis sufficiently demonstrates that the TSF behaves in accordance with the SFRs, the developer need perform no further testing of the composed TOE functionality. However, if the base component was not used in the testing of the dependent component, or the configuration of either component varied, then the developer is to perform testing of the composed TOE. This may take the form of repeating the dependent component developer testing of the dependent component, provided this adequately demonstrates the composed TOE TSF behaves in accordance with the SFRs.

1. The developer is to provide evidence of testing the base component interfaces used in the composition. The operation of base component TSFIs would have been tested as part of the [ATE](#ate) activities during evaluation of the base component. Therefore, provided the appropriate interfaces were included within the test sample of the base component evaluation and it was determined in [ACO\_COR](#aco_cor) that the base component is operating in accordance with the base component evaluated configuration, with all security functionality required by the dependent component included in the TSF, the evaluator action [ACO\_CTT.1.1E](#aco_ctt.1.1e) may be met through reuse of the base component [ATE](#ate) verdicts.
2. If this is not the case, the base component interfaces used relevant to the composition that are affected by any variations to the evaluated configuration and any additional security functionally will be tested to ensure they demonstrate the expected behaviour. The expected behaviour to be tested is that described in the [ACO\_REL](#aco_rel) evidence.

### Interface testing (ACO\_CTT.1)

Dependencies: [ACO\_REL.1](#aco_rel.1), [ACO\_DEV.1](#aco_dev.1)

Developer action elements

ACO\_CTT.1.1D The developer shall provide composed TOE test documentation.

ACO\_CTT.1.2D The developer shall provide base component interface test documentation.

ACO\_CTT.1.3D The developer shall provide the composed TOE for testing.

ACO\_CTT.1.4D The developer shall provide an equivalent set of resources to those that were used in the base component developer’s functional testing of the base component.

Content and presentation elements

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.

ACO\_CTT.1.2C The test documentation from the developer execution of the composed TOE tests shall demonstrate that the TSF behaves as specified.

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.

ACO\_CTT.1.4C The base component shall be suitable for testing.

Evaluator action elements

ACO\_CTT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACO\_CTT.1.2E The evaluator shall execute a sample of test in the test documentation to verify the developer test results.

ACO\_CTT.1.3E The evaluator shall test a subset of the TSF interfaces of the composed TOE to confirm that the composed TSF operates as specified.

### Rigorous interface testing (ACO\_CTT.2)

Dependencies: [ACO\_REL.2](#aco_rel.2), [ACO\_DEV.2](#aco_dev.2)

Developer action elements

ACO\_CTT.2.1D The developer shall provide composed TOE test documentation.

ACO\_CTT.2.2D The developer shall provide base component interface test documentation.

ACO\_CTT.2.3D The developer shall provide the composed TOE for testing.

ACO\_CTT.2.4D The developer shall provide an equivalent set of resources to those that were used in the base component developer’s functional testing of the base component.

Content and presentation elements

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.

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.**

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.**

ACO\_CTT.2.4C The base component shall be suitable for testing.

Evaluator action elements

ACO\_CTT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACO\_CTT.2.2E The evaluator shall execute a sample of test in the test documentation to verify the developer test results.

ACO\_CTT.2.3E The evaluator shall test a subset of the TSF interfaces of the composed TOE to confirm that the composed TSF operates as specified.

## Development evidence (ACO\_DEV)

Objectives

1. This family sets out requirements for a specification of the base component in increasing levels of detail. Such information is required to gain confidence that the appropriate security functionality is provided to support the requirements of the dependent component (as identified in the reliance information).

Levelling criteria

1. The components are levelled on the basis of increasing amounts of detail about the interfaces provided, and how they are implemented.

Application notes

1. The TSF of the base component is often defined without knowledge of the dependencies of the possible applications with which it may by composed. The TSF of this base component is defined to include all parts of the base component that have to be relied upon for enforcement of the base component SFRs. This will include all parts of the base component required to implement the base component SFRs.
2. The functional specification of the base component will describe the TSFI in terms of the interfaces the base component provides to allow an external entity to invoke operations of the TSF. This includes interfaces to the human user to permit interaction with the operation of the TSF invoking SFRs and also interfaces allowing an external IT entity to make calls into the TSF.
3. The functional specification only provides a description of what the TSF provides at its interface and the means by which that TSF functionality are invoked. Therefore, the functional specification does not necessarily provide a complete interface specification of all possible interfaces available between an external entity and the base component. It does not include what the TSF expects/requires from the operational environment. The description of what a dependent component TSF relies upon of a base component is considered in [ACO\_REL](#aco_rel) and the development information evidence provides a response to the interfaces specified.
4. The development information evidence includes a specification of the base component. This may be the evidence used during evaluation of the base component to satisfy the [ADV](#adv) requirements, or may be another form of evidence produced by either the base component developer or the composed TOE developer. This specification of the base component is used during [ACO\_DEV](#aco_dev) to gain confidence that the appropriate security functionality is provided to support the requirements of the dependent component. The level of detail required of this evidence increases to reflect the level of required assurance in the composed TOE. This is expected to broadly reflect the increasing confidence gained from the application of the assurance packages to the components. The evaluator determines that this description of the base component is consistent with the reliance information provided for the dependent component.

### Functional Description (ACO\_DEV.1)

Dependencies: [ACO\_REL.1](#aco_rel.1)

Developer action elements

ACO\_DEV.1.1D The developer shall provide development information for the base component.

Content and presentation elements

ACO\_DEV.1.1C The development information shall describe the purpose of each interface of the base component used in the composed TOE.

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.

Evaluator action elements

ACO\_DEV.1.1E The evaluator shall confirm that the information meets all requirements for content and presentation of evidence.

ACO\_DEV.1.2E The evaluator shall determine that the interface description provided is consistent with the reliance information provided for the dependent component.

### Basic evidence of design (ACO\_DEV.2)

Dependencies: [ACO\_REL.1](#aco_rel.1)

Developer action elements

ACO\_DEV.2.1D The developer shall provide development information for the base component.

Content and presentation elements

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.

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.

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.

Evaluator action elements

ACO\_DEV.2.1E The evaluator shall confirm that the information meets all requirements for content and presentation of evidence.

ACO\_DEV.2.2E The evaluator shall determine that the interface description provided is consistent with the reliance information provided for the dependent component.

### Detailed evidence of design (ACO\_DEV.3)

Dependencies: [ACO\_REL.2](#aco_rel.2)

Developer action elements

ACO\_DEV.3.1D The developer shall provide development information for the base component.

Content and presentation elements

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.

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.

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.

ACO\_DEV.3.4C The development information shall provide a mapping from the interfaces to the subsystems of the base component.

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.

Evaluator action elements

ACO\_DEV.3.1E The evaluator shall confirm that the information meets all requirements for content and presentation of evidence.

ACO\_DEV.3.2E The evaluator shall determine that the interface description provided is consistent with the reliance information provided for the dependent component.

## Reliance of dependent component (ACO\_REL)

Objectives

1. The purpose of this family is to provide evidence that describes the reliance that a dependent component has upon the base component. This information is useful to persons responsible for integrating the component with other evaluated IT components to form the composed TOE, and for providing insight into the security properties of the resulting composition.
2. This provides a description of the interface between the dependent and base components of the composed TOE that may not have been analysed during evaluation of the individual components, as the interfaces were not TSFIs of the individual component TOEs.

Levelling criteria

1. The components in this family are levelled according to the amount of detail provided in the description of the reliance by the dependent component upon the base component.

Application notes

1. The [ACO\_REL](#aco_rel) family considers the interactions between the components where the dependent component relies upon a service from the base component to support the operation of security functionality of the dependent component. The interfaces into these services of the base component may not have been considered during evaluation of the base component because the service in the base component was not considered security-relevant in the component evaluation, either because of the inherent purpose of the service (e.g. adjust type font) or because associated ISO/IEC 15408-2 SFRs are not being claimed in the base component’s ST (e.g. the login interface when no FIA: Identification and authentication SFRs are claimed). These interfaces into the base component are often viewed as functional interfaces in the evaluation of the base component, and are in addition to the security interfaces (TSFI) considered in the functional specification.
2. In summary, the TSFIs described in the functional specification only include the calls made into a TSF by external entities and responses to those calls. Calls made by a TSF, which were not explicitly considered during evaluation of the components, are described by the reliance information provided to satisfy [ACO\_REL](#aco_rel).

### Basic reliance information (ACO\_REL.1)

Developer action elements

ACO\_REL.1.1D The developer shall provide reliance information of the dependent component.

Content and presentation elements

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.

ACO\_REL.1.2C The reliance information shall describe all interactions through which the dependent component TSF requests services from the base component.

ACO\_REL.1.3C The reliance information shall describe how the dependent TSF protects itself from interference and tampering by the base component.

Evaluator action elements

ACO\_REL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Reliance information (ACO\_REL.2)

Developer action elements

ACO\_REL.2.1D The developer shall provide reliance information of the dependent component.

Content and presentation elements

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.

ACO\_REL.2.2C The reliance information shall describe all interactions through which the dependent component TSF requests services from the base component.

ACO\_REL.2.3C The reliance information shall describe each interaction in terms of the interface used and the return values from those interfaces.

ACO\_REL.2.4C The reliance information shall describe how the dependent TSF protects itself from interference and tampering by the base component.

Evaluator action elements

ACO\_REL.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Composition vulnerability analysis (ACO\_VUL)

Objectives

1. This family calls for an analysis of vulnerability information available in the public domain and of vulnerabilities that may be introduced as a result of the composition.

Levelling criteria

1. The components in this family are levelled on the basis of increasing scrutiny of vulnerability information from the public domain and independent vulnerability analysis.

Application notes

1. The developer will provide details of any residual vulnerabilities reported during evaluation of the components. These may be gained from the component developers or evaluation reports for the components. These will be used as inputs into the evaluator’s vulnerability analysis of the composed TOE in the operational environment.
2. The operational environment of the composed TOE is examined to ensure that the assumptions and objectives for the component operational environment (specified in each component ST) are satisfied in the composed TOE. An initial analysis of the consistency of assumptions and objectives between the components and the composed TOE STs will have been performed during the conduct of the [ASE](#ase) activities for the composed TOE. However, this analysis is revisited with the knowledge acquired during the [ACO\_REL](#aco_rel), [ACO\_DEV](#aco_dev) and the [ACO\_COR](#aco_cor) activities to ensure that, for example, assumptions of the dependent component that were addressed by the environment in the dependent component ST are not reintroduced as a result of composition (i.e. that the base component adequately addresses the assumptions of the dependent component ST in the composed TOE).
3. A search by the evaluator for issues in each component will identify potential vulnerabilities reported in the public domain since completion of the evaluation of the components. Any potential vulnerabilities will then be subject to testing.
4. If the base component used in the composed TOE has been the subject of assurance continuity activities since certification, the evaluator will consider during the composed TOE vulnerability analysis activities the changes made in base component.

### Composition vulnerability review (ACO\_VUL.1)

Dependencies: [ACO\_DEV.1](#aco_dev.1)

Developer action elements

ACO\_VUL.1.1D The developer shall provide the composed TOE for testing.

Content and presentation elements

ACO\_VUL.1.1C The composed TOE shall be suitable for testing.

Evaluator action elements

ACO\_VUL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACO\_VUL.1.2E The evaluator shall perform an analysis to determine that any residual vulnerabilities identified for the base and dependent components are not exploitable in the composed TOE in its operational environment.

ACO\_VUL.1.3E The evaluator shall perform a search of public domain sources to identify possible vulnerabilities arising from use of the base and dependent components in the composed TOE operational environment.

ACO\_VUL.1.4E The evaluator shall conduct penetration testing, based on the identified vulnerabilities, to demonstrate that the composed TOE is resistant to attacks by an attacker with basic attack potential.

### Composition vulnerability analysis (ACO\_VUL.2)

Dependencies: [ACO\_DEV.2](#aco_dev.2)

Developer action elements

ACO\_VUL.2.1D The developer shall provide the composed TOE for testing.

Content and presentation elements

ACO\_VUL.2.1C The composed TOE shall be suitable for testing.

Evaluator action elements

ACO\_VUL.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACO\_VUL.2.2E The evaluator shall perform an analysis to determine that any residual vulnerabilities identified for the base and dependent components are not exploitable in the composed TOE in its operational environment.

ACO\_VUL.2.3E The evaluator shall perform a search of public domain sources to identify possible vulnerabilities arising from use of the base and dependent components in the composed TOE operational environment.

ACO\_VUL.2.4E The evaluator shall perform an independent vulnerability analysis of the composed TOE, using the guidance documentation, reliance information and composition rationale to identify potential vulnerabilities in the composed TOE.

ACO\_VUL.2.5E The evaluator shall conduct penetration testing, based on the identified vulnerabilities, to demonstrate that the composed TOE is resistant to attacks by an attacker with basic attack potential.

### Enhanced-Basic Composition vulnerability analysis (ACO\_VUL.3)

Dependencies: [ACO\_DEV.3](#aco_dev.3)

Developer action elements

ACO\_VUL.3.1D The developer shall provide the composed TOE for testing.

Content and presentation elements

ACO\_VUL.3.1C The composed TOE shall be suitable for testing.

Evaluator action elements

ACO\_VUL.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ACO\_VUL.3.2E The evaluator shall perform an analysis to determine that any residual vulnerabilities identified for the base and dependent components are not exploitable in the composed TOE in its operational environment.

ACO\_VUL.3.3E The evaluator shall perform a search of public domain sources to identify possible vulnerabilities arising from use of the base and dependent components in the composed TOE operational environment.

ACO\_VUL.3.4E The evaluator shall perform an independent vulnerability analysis of the composed TOE, using the guidance documentation, reliance information and composition rationale to identify potential vulnerabilities in the composed TOE.

ACO\_VUL.3.5E The evaluator shall conduct penetration testing, based on the identified vulnerabilities, to demonstrate that the composed TOE is resistant to attacks by an attacker with **Enhanced-Basic** attack potential.

# Class ADV Development

## Introduction

The requirements of the Development class provide information about the TOE. The knowledge obtained by this information is used as the basis for conducting vulnerability analysis and testing upon the TOE, as described in the [AVA](#ava) and [ATE](#ate) classes.

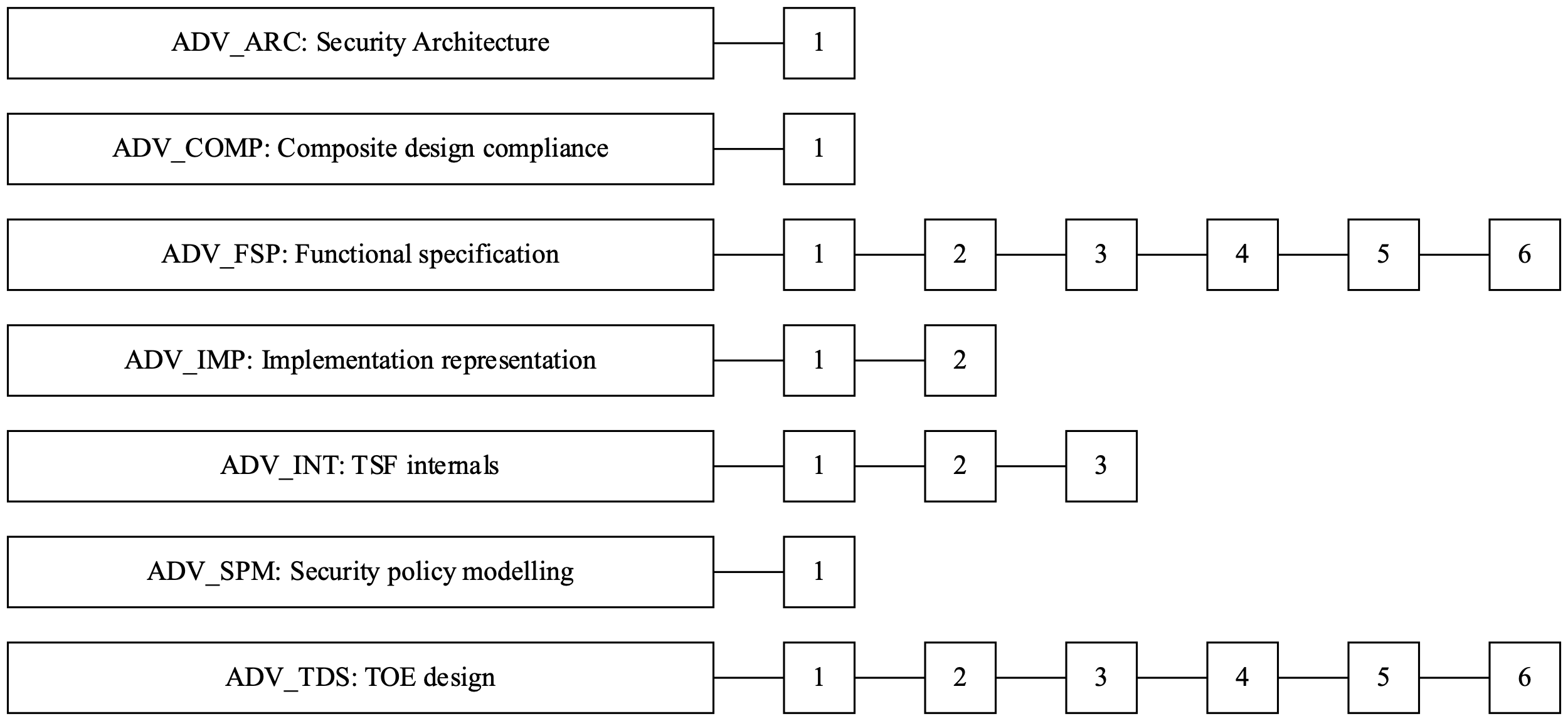
1. The Development class encompasses seven families of requirements for structuring and representing the TSF at various levels and varying forms of abstraction. These families include:

* requirements for the description (at the various levels of abstraction) of the design and implementation of the SFRs ([ADV\_FSP](#adv_fsp), [ADV\_TDS](#adv_tds), [ADV\_IMP](#adv_imp) and [ADV\_COMP](#adv_comp)).
* requirements for the description of the architecture-oriented features of domain separation, TSF self-protection and non-bypassability of the security functionality ([ADV\_ARC](#adv_arc)).
* requirements for a security policy model and for correspondence mappings between security policy model and the functional specification ([ADV\_SPM](#adv_spm)).
* requirements on the internal structure of the TSF, which covers aspects such as modularity, layering, and minimization of complexity ([ADV\_INT](#adv_int)).

1. When documenting the security functionality of a TOE, there are two properties that need to be demonstrated. The first property is that the security functionality works correctly, i.e. it performs as specified. The second property, and one that is arguably harder to demonstrate, is that the TOE cannot be used in a way such that the security functionality can be corrupted or bypassed. These two properties require somewhat different approaches in analysis, and so the families in [ADV](#adv) are structured to support these different approaches.
2. The families [ADV\_FSP](#adv_fsp), [ADV\_TDS](#adv_tds), [ADV\_IMP](#adv_imp), and [ADV\_SPM](#adv_spm) deal with the first property: the specification of the security functionality. The families [ADV\_ARC](#adv_arc) and [ADV\_INT](#adv_int) deal with the second property: the specification of the design of the TOE demonstrating the security functionality cannot be corrupted or bypassed. It should be noted that both properties need to be realized: the more confidence one has that the properties are satisfied, the more trustworthy the TOE is.
3. The TSF of a composite product are represented at various levels of abstraction in the families of the development class [ADV](#adv). The family [ADV\_COMP](#adv_comp) determines whether the requirements on the dependent component, imposed by the related base componen, are fulfilled in a composite product.
4. Due to the distribution of the TSF of a composite product to various levels in the families of the class [ADV](#adv), this family is not represented in Figure 7. The components in the families are designed so that more assurance can be gained as the components hierarchically increase.
5. The paradigm for the families targeted at the first property is one of design decomposition. At the highest level, there is a functional specification of the TSF in terms of its interfaces (describing what the TSF does in terms of requests to the TSF for services and resulting responses), decomposing the TSF into smaller units (dependent on the assurance desired and the complexity of the TOE) and describing how the TSF accomplishes its functions (to a level of detail commensurate with the assurance level), and showing the implementation of the TSF. A formal model of the security behaviour also may be given. All levels of decomposition are used in determining the completeness and accuracy of all other levels, ensuring that the levels are mutually supportive. The requirements for the various TSF representations are separated into different families, to allow the PP/ST author to specify which TSF representations are required. The level chosen will dictate the assurance desired/gained.
6. Figure 7 indicates the relationships among the various TSF representations of the [ADV](#adv) class, as well as their relationships with other classes. As the figure indicates, the [APE](#ape) and [ASE](#ase) classes define the requirements for the correspondence between the SFRs and the security objectives for the TOE. Class [ASE](#ase) also defines requirements for the correspondence between both the security objectives and SFRs, and for the TOE summary specification which explains how the TOE meets its SFRs. The activities of [ALC\_CMC.5.2E](#alc_cmc.5.2e) include the verification that the TSF that is tested under the [ATE](#ate) and [AVA](#ava) classes is in fact the one described by all of the [ADV](#adv) decomposition levels.
7. The requirements for all other correspondence shown in Figure 7 are defined in the [ADV](#adv) class for the TOE. The [ADV\_SPM](#adv_spm) family defines the requirements for formally modelling selected SFRs and providing correspondence between the functional specification and the formal model. Each assurance family specific to a TSF representation (i.e. [ADV\_FSP](#adv_fsp), [ADV\_TDS](#adv_tds) and [ADV\_IMP](#adv_imp)) defines requirements relating that TSF representation to the SFRs. All decompositions must accurately reflect all other decompositions (i.e. be mutually supportive); the developer supplies the tracings in the last .C elements of the components. Assurance relating to this factor is obtained during the analysis for each of the levels of decomposition by referring to other levels of decomposition (in a recursive fashion) while the analysis of a particular level of decomposition is being performed; the evaluator verifies the correspondence as part of the second E element. The understanding gained from these levels of decomposition form the basis of the functional and penetration testing efforts.
8. The [ADV\_INT](#adv_int) family is not represented in this figure, as it is related to the internal structure of the TSF, and is only indirectly related to the process of refinement of the TSF representations. Similarly, the [ADV\_ARC](#adv_arc) family is not represented in the figure because it relates to the architectural soundness, rather than representation, of the TSF. Both [ADV\_INT](#adv_int) and [ADV\_ARC](#adv_arc) relate to the analysis of the property that the TOE cannot be made to circumvent or corrupt its security functionality.
9. The TOE security functionality (TSF) consists of all parts of the TOE that shall be relied upon for enforcement of the SFRs. The TSF includes both functionality that directly enforces the SFRs, as well as functionality that, while not directly enforcing the SFRs, contributes to their enforcement in a more indirect manner, including functionality with the capability to cause the SFRs to be violated. This includes portions of the TOE that are invoked on start-up that are responsible for putting the TSF into its initial secure state.
10. Several important concepts were used in the development of the components of the [ADV](#adv) families. These concepts, while introduced briefly here, are explained more fully in the application notes for the families.
11. One over-riding notion is that, as more information becomes available, greater assurance can be obtained that the security functionality
12. is correctly implemented;
13. cannot be corrupted; and
14. cannot be bypassed.

This is done through the verification that the documentation is correct and consistent with other documentation, and by providing information that can be used to ensure that the testing activities (both functional and penetration testing) are comprehensive. This is reflected in the levelling of the components of the families. In general, components are levelled based on the amount of information that is to be provided (and subsequently analysed).

1. While not true for all TOEs, it is generally the case that the TSF is sufficiently complex that there are portions of the TSF that deserve more intense examination than other portions of the TSF. Determining those portions is unfortunately somewhat subjective, thus terminology and components have been defined such that as the level of assurance increases, the responsibility for determining what portions of the TSF need to be examined in detail shifts from the developer to the evaluator. To aid in expressing this concept, the following terminology is introduced. It should be noted that in the families of the class, this terminology is used when expressing SFR-related portions of the TOE (i.e. elements and work units embodied in the [ADV\_FSP](#adv_fsp), [ADV\_TDS](#adv_tds), and [ADV\_IMP](#adv_imp) families). While the general concept (that some portions of the TOE are more interesting than others) applies to other families, the criteria are expressed differently in order to obtain the assurance required.
2. All portions of the TSF are security relevant, meaning that they must preserve the security of the TOE as expressed by the SFRs and requirements for domain separation and non-bypassability. One aspect of security relevance is the degree to which a portion of the TSF enforces a security requirement. Since different portions of the TOE play different roles (or no apparent role at all) in enforcing security requirements, this creates a continuum of SFR relevance: at one end of this continuum are portions of the TOE that are termed SFR-enforcing.
3. Such portions play a direct role in implementing any SFR on the TOE. Such SFRs refer to any functionality provided by one of the SFRs contained in the ST. It should be noted that the definition of plays a role in for SFR-enforcing functionality is impossible to express quantitatively.
4. For example, in the implementation of a Discretionary Access Control (DAC) mechanism, a very narrow view of SFR-enforcing can be the several lines of code that actually perform the check of a subject’s attributes against the object’s attributes.
5. A broader view would include the software entity (e.g. C function) that contained the several lines of code. A broader view still would include callers of the C function, since they would be responsible for enforcing the decision returned by the attribute check.
6. A still broader view would include any code in the call tree (or programming equivalent for the implementation language used) for that C function (e.g. a sort function that sorted access control list entries in a first-match algorithm implementation). At some point, the component is not so much enforcing the security policy but rather plays a supporting role; such components are termed SFR supporting. One of the characteristics of SFR-supporting functionality is that it is trusted to preserve the correctness of the SFR implementation by operating without error.
7. Such functionality may be depended on by SFR-enforcing functionality, but the dependence is generally at a functional level; for example, memory management, buffer management, etc.
8. Further down on the security relevance continuum is functionality termed SFR non-interfering. Such functionality has no role in implementing the SFRs and is likely part of the TSF because of its environment; for example, any code running in a privileged hardware mode on an operating system. It needs to be considered part of the TSF because, if compromised (or replaced by malicious code), it can compromise the correct operation of an SFR by virtue of its operating in the privileged hardware mode.
9. An example of SFR non-interfering functionality can be a set of mathematical floating point operations implemented in kernel mode for speed considerations.
10. The family [ADV\_ARC](#adv_arc) provides for requirements and analysis of the TOE based on properties of domain separation, self-protection, and non-bypassability. These properties relate to the SFRs in that, if these properties are not present, it will likely lead to the failure of mechanisms implementing SFRs. Functionality and design relating to these properties is not considered a part of the continuum described above, but instead is treated separately due to its fundamentally different nature and analysis requirements.
11. The difference in analysis of the implementation of SFRs (SFR-enforcing and SFR-supporting functionality) and the implementation of somewhat fundamental security properties of the TOE, which include the initialisation, self-protection, and non-bypassability concerns, is that the SFR-related functionality is more or less directly visible and relatively easy to test, while the above-mentioned properties require varying degrees of analysis on a much broader set of functionality. Further, the depth of analysis for such properties will vary depending on the design of the TOE. The [ADV](#adv) families are constructed to address this by a separate family ([ADV\_ARC](#adv_arc)) devoted to analysis of the initialisation, self-protection, and non-bypassability requirements, while the other families are concerned with analysis of the functionality supporting SFRs.
12. Even in cases where different descriptions are necessary for the multiple levels of abstraction, it is not absolutely necessary for each and every TSF representation to be in a separate document. Indeed, it may be the case that a single document meets the documentation requirements for more than one TSF representation, since it is the information about each of these TSF representations that is required, rather than the resulting document structure. In cases where multiple TSF representations are combined within a single document, the developer should indicate which portions of the documents meet which requirements.
13. Three types of specification style are mandated by this class: informal, semiformal and formal. The functional specification and TOE design documentation are always written in either informal or semiformal style. A semiformal style reduces the ambiguity in these documents over an informal presentation. A formal specification may also be required in addition to the semi-formal presentation; the value is that a description of the TSF in more than one way will add increased assurance that the TSF has been completely and accurately specified.
14. An informal specification is written as prose in natural language. Natural language is used here as meaning communication in any commonly spoken tongue (e.g. Spanish, German, French, English, Dutch). An informal specification is not subject to any notational or special restrictions other than those required as ordinary conventions for that language (e.g. grammar and syntax). While no notational restrictions apply, the informal specification is also required to provide defined meanings for terms that are used in a context other than that accepted by normal usage.
15. The difference between semiformal and informal documents is only a matter of formatting or presentation: a semiformal notation includes, e.g. an explicit glossary of terms, a standardised presentation format. A semiformal specification is written to a standard presentation template. The presentation should use terms consistently if written in a natural language. The presentation may also use more structured languages/diagrams (e.g. data-flow diagrams, state transition diagrams, entity-relationship diagrams, data structure diagrams, and process or program structure diagrams). Whether based on diagrams or natural language, a set of conventions must be used in the presentation. The glossary explicitly identifies the words that are being used in a precise and constant manner; similarly, the standardised format implies that extreme care has been taken in methodically preparing the document in a manner that maximises clarity. It should be noted that fundamentally different portions of the TSF may have different semiformal notation conventions and presentation styles (as long as the number of different “semiformal notations” is small); this still conforms to the concept of a semiformal presentation.
16. A formal specification is written in a notation based upon well-established mathematical concepts and is typically accompanied by supporting explanatory (informal) prose. These mathematical concepts are used to define the syntax and semantics of the notation and the proof rules that support logical reasoning. The syntactic and semantic rules supporting a formal notation should define how to recognize constructs unambiguously and determine their meaning. There needs to be evidence that it is impossible to derive contradictions, and all rules supporting the notation need to be defined or referenced.



|  | [ADV\_FSP.1](#adv_fsp.1) | [ADV\_FSP.2](#adv_fsp.2) | [ADV\_FSP.3](#adv_fsp.3) | [ADV\_FSP.4](#adv_fsp.4) | [ADV\_FSP.5](#adv_fsp.5) | [ADV\_FSP.6](#adv_fsp.6) | [ADV\_IMP.1](#adv_imp.1) | [ADV\_TDS.1](#adv_tds.1) | [ADV\_TDS.3](#adv_tds.3) | [ALC\_CMC.5](#alc_cmc.5) | [ALC\_TAT.1](#alc_tat.1) | [ASE\_OBJ.2](#ase_obj.2) | [ASE\_REQ.2](#ase_req.2) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**ADV\_ARC.1**](#adv_arc.1) | X | - |  |  |  |  |  | X |  |  |  |  |  |
| [**ADV\_COMP.1**](#adv_comp.1) |  |  |  |  |  |  |  |  |  |  |  |  |  |
| [**ADV\_FSP.1**](#adv_fsp.1) |  |  |  |  |  |  |  |  |  |  |  |  |  |
| [**ADV\_FSP.2**](#adv_fsp.2) |  | - |  |  |  |  |  | X |  |  |  |  |  |
| [**ADV\_FSP.3**](#adv_fsp.3) |  | - |  |  |  |  |  | X |  |  |  |  |  |
| [**ADV\_FSP.4**](#adv_fsp.4) |  | - |  |  |  |  |  | X |  |  |  |  |  |
| [**ADV\_FSP.5**](#adv_fsp.5) |  | - |  | - |  |  | X | X | - |  | - |  |  |
| [**ADV\_FSP.6**](#adv_fsp.6) |  | - |  | - |  |  | X | X | - |  | - |  |  |
| [**ADV\_IMP.1**](#adv_imp.1) |  | - |  | - |  |  |  | - | X |  | X |  |  |
| [**ADV\_IMP.2**](#adv_imp.2) |  | - |  | - |  |  |  | - | X | X | X |  |  |
| [**ADV\_INT.1**](#adv_int.1) |  | - |  | - |  |  | X | - | X |  | X |  |  |
| [**ADV\_INT.2**](#adv_int.2) |  | - |  | - |  |  | X | - | X |  | X |  |  |
| [**ADV\_INT.3**](#adv_int.3) |  | - |  | - |  |  | X | - | X |  | X |  |  |
| [**ADV\_SPM.1**](#adv_spm.1) |  | - |  | X |  |  |  | - |  |  |  | X | X |
| [**ADV\_TDS.1**](#adv_tds.1) |  | X |  |  |  |  |  | - |  |  |  |  |  |
| [**ADV\_TDS.2**](#adv_tds.2) |  | - | X |  |  |  |  | - |  |  |  |  |  |
| [**ADV\_TDS.3**](#adv_tds.3) |  | - |  | X |  |  |  | - |  |  |  |  |  |
| [**ADV\_TDS.4**](#adv_tds.4) |  | - |  | - | X |  | - | - | - |  | - |  |  |
| [**ADV\_TDS.5**](#adv_tds.5) |  | - |  | - | X |  | - | - | - |  | - |  |  |
| [**ADV\_TDS.6**](#adv_tds.6) |  | - |  | - |  | X | - | - | - |  | - |  |  |

## Security Architecture (ADV\_ARC)

Objectives

1. The objective of this family is for the developer to provide a description of the security architecture of the TSF. This will allow analysis of the information that, when coupled with the other evidence presented for the TSF, will confirm the TSF achieves the desired properties. The security architecture descriptions support the implicit claim that security analysis of the TOE can be achieved by examining the TSF; without a sound architecture, the entire TOE functionality would have to be examined.

Levelling criteria

1. This family contains only one component.

Application notes

1. The properties of self-protection, domain separation, and non-bypassability are distinct from security functionality expressed by ISO/IEC 15408-2 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 TSF and enforced by the correct implementation of that design.
2. The approach used in this family is for the developer to design and provide a TSF that exhibits the above-mentioned properties, and to provide evidence (in the form of documentation) that explains these properties of the TSF. This explanation is provided at the same level of detail as the description of the SFR-enforcing elements of the TOE in the TOE design document. The evaluator has the responsibility for looking at the evidence and, coupled with other evidence delivered for the TOE and TSF, determining that the properties are achieved.
3. Specification of security functionality implementing the SFRs in [ADV\_FSP](#adv_fsp) and [ADV\_TDS](#adv_tds) will not necessarily describe mechanisms employed in implementing self-protection and non-bypassability (e.g. memory management mechanisms). Therefore, the material needed to provide the assurance that these requirements are being achieved is better suited to a presentation separate from the design decomposition of the TSF as embodied in [ADV\_FSP](#adv_fsp) and [ADV\_TDS](#adv_tds). This is not to imply that the security architecture description called for by this component cannot reference or make use of the design decomposition material; but it is likely that much of the detail present in the decomposition documentation will not be relevant to the argument being provided for the security architecture description document.
4. The description of architectural soundness can be thought of as a developer’s vulnerability analysis, in that it provides the justification for why the TSF is sound and enforces all of its SFRs. Where the soundness is achieved through specific security mechanisms, these will be tested as part of the [ATE\_DPT](#ate_dpt) requirements; where the soundness is achieved solely through the architecture, the behaviour will be tested as part of the [AVA](#ava) requirements.
5. This family consists of requirements for a security architecture description that describes the self-protection, domain separation, non-bypassability principles, including a description of how these principles are supported by the parts of the TOE that are used for TSF initialisation.
6. In case of a multi-assurance evaluation the properties of self-protection, domain separation, and non-bypassability may also be described for boundaries between the sub-TSF.
7. Additional information on the security architecture properties of self-protection, domain separation, and non-bypassability can be found in A.1, [ADV\_ARC](#adv_arc): Supplementary material on security architectures.

### Security architecture description (ADV\_ARC.1)

Dependencies: [ADV\_FSP.1](#adv_fsp.1), [ADV\_TDS.1](#adv_tds.1)

Developer action elements

ADV\_ARC.1.1D The developer shall design and implement the TOE so that the security features of the TSF cannot be bypassed.

ADV\_ARC.1.2D The developer shall design and implement the TSF so that it is able to protect itself from tampering by untrusted active entities.

ADV\_ARC.1.3D The developer shall provide a security architecture description of the TSF.

Content and presentation elements

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.

ADV\_ARC.1.2C The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.

ADV\_ARC.1.3C The security architecture description shall describe how the TSF initialisation process is secure.

ADV\_ARC.1.4C The security architecture description shall demonstrate that the TSF protects itself from tampering.

ADV\_ARC.1.5C The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.

Evaluator action elements

ADV\_ARC.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Composite design compliance (ADV\_COMP)

Objectives

1. The aim of this family is to determine whether the requirements on the dependent component, imposed by the related base component, are fulfilled in the composite product.

Levelling criteria

1. This family contains only one component.

Application notes

1. 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 (in form of observations and recommendations) and report of the base component evaluation authority (e.g. in form of constraints 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 fulfilled. The composite product evaluator shall verify that all stipulations for the dependent component that are imposed by the base component and provided in its evaluation related documentation are fulfilled by the composite product, i.e. have been taken into account by the dependent component developer.
2. 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 report of the base component evaluation authority,
* a rationale for secure composite product implementation including evidence prepared by the dependent component developer.

1. The TSF of the composite product are represented at various levels of abstraction in the families of the development class [ADV](#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 [ADV\_TDS](#adv_tds), [ADV\_ARC](#adv_arc) and [ADV\_IMP](#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).
2. 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](#adv_fsp) as representation level is not appropriate for analysing the design compliance.
3. Security architecture [ADV\_ARC](#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). In the context of the [ADV\_ARC](#adv_arc), the composite product evaluator shall:
4. determine whether the dependent component uses services of the related base document within its own composite product ST to provide domain seitemtion, self-protection, non-bypassability and protected start-up; if no, there are no further composite activities for [ADV\_ARC](#adv_arc); if yes, then
5. 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).

As consistency of the composite product security policy has already been considered in the context of the ST in the assurance family [ASE\_COMP](#ase_comp), there is no necessity to consider non-contradictoriness of the security policy model ([ADV\_SPM](#adv_spm)) of the composite product and the security policy model of its related base component.

### Design compliance with the base component-related user guidance, ETR for composite evaluation and report of the base component evaluation authority (ADV\_COMP.1)

Developer action elements

ADV\_COMP.1.1D The developer shall provide a design compliance justification.

Content and presentation elements

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.

Evaluator action elements

ADV\_COMP.1.1E The evaluator shall confirm that the rationale for design compliance is complete, coherent, and internally consistent.

## Functional specification (ADV\_FSP)

Objectives

1. This family levies requirements upon the functional specification, which describes the TSF interfaces (TSFIs). The TSFI consists of all means by which external entities (or subjects in the TOE but outside of the TSF) supply data to the TSF, receive data from the TSF or invoke services from the TSF. It does not describe how the TSF processes those service requests, nor does it describe the communication when the TSF invokes services from its operational environment; this information is addressed by the [ADV\_TDS](#adv_tds) and [ACO\_REL](#aco_rel) families, respectively.
2. This family provides assurance directly by allowing the evaluator to understand how the TSF meets the claimed SFRs. It also provides assurance indirectly, as input to other assurance families and classes:

* [ADV\_ARC](#adv_arc), where the description of the TSFIs may be used to gain better understanding of how the TSF is protected against corruption (i.e. subversion of self-protection or domain separation) and/or bypass;
* [ATE](#ate), where the description of the TSFIs is an important input for both developer and evaluator testing;
* [AVA](#ava), where the description of the TSFIs is used to search for vulnerabilities.

Levelling criteria

1. The components in this family are levelled on the degree of detail required of the description of the TSFIs, and the degree of formalism required of the description of the TSFIs.

Application notes

General

1. Once the TSFIs are determined (see A.2.2 for guidance and examples of determining TSFI), they are described. At lower-level components, developers focus their documentation (and evaluators focus their analysis) on the more security-relevant aspects of the TOE. Three categories of TSFIs are defined, based upon the relevance the services available through them have to the SFRs being claimed:
2. If a service available through an interface can be traced to one of the SFRs levied on the TSF, then that interface is termed SFR-enforcing. Note that it is possible that an interface may have various services and results, some of which may be SFR-enforcing and some of which may not.
3. Interfaces to (or services available through an interface relating to) services that SFR-enforcing functionality depend upon, but need only to function correctly in order for the security policies of the TOE to be preserved, are termed SFR-supporting.
4. Interfaces to services 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 services or results. In contrast, an SFR-enforcing interface may have SFR-supporting services (for example, the ability to set the system clock may be an SFR-enforcing service of an interface, but if that same interface is used to display the system date that service may be only SFR-supporting). An example of a purely SFR-supporting interface is a system call interface that is used both by users and by a portion of the TSF that is running on behalf of users.

1. As more information about the TSFIs becomes available, the greater the assurance that can be gained that the interfaces are correctly categorised/analysed. The requirements are structured such that, at the lowest level, the information required for SFR non-interfering interfaces is the minimum necessary in order for the evaluator to make this determination in an effective manner. At higher levels, more information becomes available so that the evaluator has greater confidence in the designation.
2. The purpose in defining these labels (SFR-enforcing, SFR-supporting, and SFR-non-interfering) and for levying different requirements upon each (at the lower assurance components) is to provide a first approximation of where to focus the analysis and the evidence upon which that analysis is performed. If the developer’s documentation of the TSF interfaces describes all of the interfaces to the degree specified in the requirements for the SFR-enforcing interfaces (i.e. if the documentation exceeds the requirements), there is no need for the developer to create new evidence to match the requirements. Similarly, because the labels are merely a means of differentiating the interface types within the requirements, there is no need for the developer to update the evidence solely to label the interfaces as SFR-enforcing, SFR-supporting, and SFR-non-interfering. The primary purpose of this labelling is to allow developers with less mature development methodologies (and associated artefacts, such as detailed interface and design documentation) to provide only the necessary evidence without undue cost.
3. The last C element of each component within this family provides a direct correspondence between the SFRs and the functional specification, i.e. an indication of which interfaces are used to invoke each of the claimed SFRs. In the cases where the ST contains such functional requirements as ISO/IEC 15408-2, whose functionality may not manifest itself at the TSFIs, the functional specification and/or the tracing is expected to identify these SFRs; including them in the functional specification helps to ensure that they are not lost at lower levels of decomposition, where they will be relevant.

Detail about the interfaces

1. The requirements define collections of details about TSFI to be provided. For the purposes of the requirements, interfaces are specified (in varying degrees of detail) in terms of their purpose, method of use, parameters, parameter descriptions, and error messages.
2. The purpose of an interface is a high-level description of the general goal of the interface (e.g. process GUI commands, receive network packets, provide printer output, etc.).
3. The interface’s method of use describes how the interface is supposed to be used. This description should be built around the various interactions available at that interface. For instance, if the interface were a Unix command shell, ls, mv and cp would be interactions for that interface. For each interaction the method of use describes what the interaction does, both for behaviour seen at the interface (e.g. the programmer calling the API, the Windows users changing a setting in the registry, etc.) as well as behaviour at other interfaces (e.g. generating an audit record).
4. Parameters are explicit inputs to and outputs from an interface that control the behaviour of that interface. For example, parameters are the arguments supplied to an API; the various fields in a packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; the flags that can be set for the ls, etc. The parameters are “identified” with a simple list of what they are.
5. A parameter description tells what the parameter is in some meaningful way. For instance, an acceptable parameter description for interface foo(i) would be “parameter i is an integer that indicates the number of users currently logged in to the system”. A description such as “parameter i is an integer” is not an acceptable.
6. The description of an interface’s actions describes what the interface does. This is more detailed than the purpose in that, while the “purpose” reveals why one might want to use it, the “actions” reveals everything that it does. These actions can be related to the SFRs or not. In cases where the interface’s action is not related to SFRs, its description is said to be summarized, meaning the description merely makes clear that it is indeed not SFR-related.
7. The error message description identifies the condition that generated it, what the message is, and the meaning of any error codes. An error message is generated by the TSF to signify that a problem or irregularity of some degree has been encountered. The requirements in this family refer to different kinds of error messages:
8. a “direct” error message is a security-relevant response through a specific TSFI invocation.
9. an “indirect” error cannot be tied to a specific TSFI invocation because it results from system-wide conditions (e.g. resource exhaustion, connectivity interruptions, etc.). Error messages that are not security-relevant are also considered “indirect”.
10. “remaining” errors are any other errors, such as those that can be referenced within the code. For example, the use of condition-checking code that checks for conditions that would not logically occur (e.g. a final “else” after a list of “case” statements), would provide for generating a catch-all error message; in an operational TOE, these error messages should never be seen.

An example functional specification is provided in A.2.4.

Components of this family

1. Increasing assurance through increased completeness and accuracy in the interface specification is reflected in the documentation required from the developer as detailed in the various hierarchical components of this family.
2. At [ADV\_FSP.1](#adv_fsp.1), the only documentation required is a characterization of all TSFIs and a high-level description of SFR-enforcing and SFR-supporting TSFIs. To provide some assurance that the “important” aspects of the TSF have been correctly characterized at the TSFIs, the developer is required to provide the purpose and method of use, parameters for the SFR-enforcing and SFR-supporting TSFIs.
3. At [ADV\_FSP.2](#adv_fsp.2), the developer is required to provide the purpose, method of use, parameters, and parameter descriptions for all TSFIs. Additionally, for the SFR-enforcing TSFIs the developer shall describe the SFR-enforcing actions and direct error messages.
4. At [ADV\_FSP.3](#adv_fsp.3), the developer must now, in addition to the information required at [ADV\_FSP.2](#adv_fsp.2), provide enough information about the SFR-supporting and SFR-non-interfering actions to show that they are not SFR-enforcing. Further, the developer must now document all of the direct error messages resulting from the invocation of SFR-enforcing TSFIs.
5. At [ADV\_FSP.4](#adv_fsp.4), all TSFIs, whether SFR-enforcing, SFR-supporting or SFR-non-interfering, must be described to the same degree, including all of the direct error messages.
6. At [ADV\_FSP.5](#adv_fsp.5) Complete semi-formal functional specification with additional error information, the TSFIs descriptions also include error messages that do not result from an invocation of a TSFI.
7. At [ADV\_FSP.6](#adv_fsp.6) Complete semi-formal functional specification with additional formal specification, in addition to the information required by [ADV\_FSP.5](#adv_fsp.5), all remaining error messages are included. The developer must also provide a formal description of the TSFI. This provides an alternative view of the TSFI that may expose inconsistencies or incomplete specification.

### Basic functional specification (ADV\_FSP.1)

Developer action elements

ADV\_FSP.1.1D The developer shall provide a functional specification.

ADV\_FSP.1.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.1.1C The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supportingTSFI.

ADV\_FSP.1.2C The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV\_FSP.1.3C The functional specification shall provide rationale for the implicit categorization of interfaces as SFR-non-interfering.

ADV\_FSP.1.4C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.1.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

### Security-enforcing functional specification (ADV\_FSP.2)

Dependencies: [ADV\_TDS.1](#adv_tds.1)

Developer action elements

ADV\_FSP.2.1D The developer shall provide a functional specification.

ADV\_FSP.2.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.2.1C The functional specification shall completely represent the TSF.

ADV\_FSP.2.2C The functional specification shall describe the purpose and method of use for **all** **TSFI.**

ADV\_FSP.2.3C The functional specification shall identify **and** **describe** all parameters associated with each TSFI.

ADV\_FSP.2.4C **For** **each** **SFR-enforcing** **TSFI,** **the** functional specification shall **describe** **the** **SFR-enforcing** **actions** associated with **the** TSFI.

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.

ADV\_FSP.2.6C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.2.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

### Functional specification with complete summary (ADV\_FSP.3)

Dependencies: [ADV\_TDS.1](#adv_tds.1)

Developer action elements

ADV\_FSP.3.1D The developer shall provide a functional specification.

ADV\_FSP.3.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.3.1C The functional specification shall completely represent the TSF.

ADV\_FSP.3.2C The functional specification shall describe the purpose and method of use for all TSFI.

ADV\_FSP.3.3C The functional specification shall identify and describe all parameters associated with each TSFI.

ADV\_FSP.3.4C For each SFR-enforcing TSFI, the functional specification shall describe the SFR-enforcing actions associated with the TSFI.

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.**

ADV\_FSP.3.6C The functional specification shall summarize the SFR-supporting and SFR-non-interfering actions associated with each TSFI.

ADV\_FSP.3.7C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.3.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

### Complete functional specification (ADV\_FSP.4)

Dependencies: [ADV\_TDS.1](#adv_tds.1)

Developer action elements

ADV\_FSP.4.1D The developer shall provide a functional specification.

ADV\_FSP.4.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.4.1C The functional specification shall completely represent the TSF.

ADV\_FSP.4.2C The functional specification shall describe the purpose and method of use for all TSFI.

ADV\_FSP.4.3C The functional specification shall identify and describe all parameters associated with each TSFI.

ADV\_FSP.4.4C The functional specification shall **describe** **all** actions associated with each TSFI.

ADV\_FSP.4.5C The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.

ADV\_FSP.4.6C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.4.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

### Complete semi-formal functional specification with additional error information (ADV\_FSP.5)

Dependencies: [ADV\_TDS.1](#adv_tds.1), [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ADV\_FSP.5.1D The developer shall provide a functional specification.

ADV\_FSP.5.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.5.1C The functional specification shall completely represent the TSF.

ADV\_FSP.5.2C The functional specification shall describe the TSFI using a semi-formal style.

ADV\_FSP.5.3C The functional specification shall describe the purpose and method of use for all TSFI.

ADV\_FSP.5.4C The functional specification shall identify and describe all parameters associated with each TSFI.

ADV\_FSP.5.5C The functional specification shall describe all actions associated with each TSFI.

ADV\_FSP.5.6C The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.

ADV\_FSP.5.7C The functional specification shall describe all error messages that do not result from an invocation of a TSFI.

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.

ADV\_FSP.5.9C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.5.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

### Complete semi-formal functional specification with additional formal specification (ADV\_FSP.6)

Dependencies: [ADV\_TDS.1](#adv_tds.1), [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ADV\_FSP.6.1D The developer shall provide a functional specification.

ADV\_FSP.6.2D The developer shall provide a formal presentation of the functional specification of the TSF.

ADV\_FSP.6.3D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation elements

ADV\_FSP.6.1C The functional specification shall completely represent the TSF.

ADV\_FSP.6.2C The functional specification shall describe the TSFI using a **formal** style.

ADV\_FSP.6.3C The functional specification shall describe the purpose and method of use for all TSFI.

ADV\_FSP.6.4C The functional specification shall identify and describe all parameters associated with each TSFI.

ADV\_FSP.6.5C The functional specification shall describe all actions associated with each TSFI.

ADV\_FSP.6.6C The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.

ADV\_FSP.6.7C The functional specification shall describe all error messages **contained** **in** **the** **TSF** **implementation** **representation.**

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.

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.

ADV\_FSP.6.10C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements

ADV\_FSP.6.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_FSP.6.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

## Implementation representation (ADV\_IMP)

Objectives

1. The function of the [ADV\_IMP](#adv_imp) family is for the developer to make available the implementation representation (and, at higher levels, the implementation itself) of the TOE in a form that can be analysed by the evaluator. The implementation representation is used in analysis activities for other families (analysing the TOE design, for instance) to demonstrate that the TOE conforms its design and to provide a basis for analysis in other areas of the evaluation (e.g. the search for vulnerabilities). The implementation representation is expected to be in a form that captures the detailed internal workings of the TSF. This may be software source code, firmware source code, hardware diagrams and/or IC hardware design language code or layout data.

Levelling criteria

1. The components in this family are levelled on the amount of implementation that is mapped to the TOE design description.

Application notes

1. 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. It is important to note that while the implementation representation must be made available to the evaluator, this does not imply that the evaluator needs to possess that representation. For instance, the developer may require that the evaluator review the implementation representation at a site of the developer’s choosing.
2. 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. The implementation representation is made available to allow analysis of other TOE design decompositions (e.g. functional specification, TOE design), and to gain confidence that the security functionality described at a higher level in the design actually appear to be implemented in the TOE. 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. For this reason, it is important that such “extra” information or related tools (e.g. scripts, compilers, etc.) be provided so that the implementation representation can be accurately determined.
3. The purpose of the mapping between the implementation representation and the TOE design description is to aid the evaluator’s analysis. The internal workings of the TOE may be better understood when the TOE design is analysed with corresponding portions of the implementation representation. The mapping serves as an index into the implementation representation. At the lower component, only a subset of the implementation representation is mapped to the TOE design description. Because of the uncertainty of which portions of the implementation representation will need such a mapping, the developer may choose either to map the entire implementation representation beforehand, or to wait to see which portions of the implementation representation the evaluator requires to be mapped.
4. 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 used by the developer, 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 maximize the effectiveness of the evaluator’s analysis efforts.
5. 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 the TOE developer taking a version of the implementation representation 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 functionality.

### Implementation representation of the TSF (ADV\_IMP.1)

Dependencies: [ADV\_TDS.3](#adv_tds.3), [ALC\_TAT.1](#alc_tat.1)

Developer action elements

ADV\_IMP.1.1D The developer shall make available the implementation representation for the entire TSF.

ADV\_IMP.1.2D The developer shall provide a mapping between the TOE design description and the sample of the implementation representation.

Content and presentation elements

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.

ADV\_IMP.1.2C The implementation representation shall be in the form used by the development personnel.

ADV\_IMP.1.3C The mapping between the TOE design description and the sample of the implementation representation shall demonstrate their correspondence.

Evaluator action elements

ADV\_IMP.1.1E The evaluator shall confirm that, for the selected sample of the implementation representation, the information provided meets all requirements for content and presentation of evidence.

### Complete mapping of the implementation representation of the TSF (ADV\_IMP.2)

Dependencies: [ADV\_TDS.3](#adv_tds.3), [ALC\_TAT.1](#alc_tat.1), [ALC\_CMC.5](#alc_cmc.5)

Developer action elements

ADV\_IMP.2.1D The developer shall make available the implementation representation for the entire TSF.

ADV\_IMP.2.2D The developer shall provide a mapping between the TOE design description and the **entire** implementation representation.

Content and presentation elements

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.

ADV\_IMP.2.2C The implementation representation shall be in the form used by the development personnel.

ADV\_IMP.2.3C The mapping between the TOE design description and the **entire** implementation representation shall demonstrate their correspondence.

Evaluator action elements

ADV\_IMP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## TSF internals (ADV\_INT)

Objectives

1. This family addresses the assessment of the internal structure of the TSF. A TSF whose internals are well-structured is easier to implement and less likely to contain flaws that can lead to vulnerabilities; it is also easier to maintain without the introduction of flaws.

Levelling criteria

1. The components in this family are levelled on the basis of the amount of structure and minimization of complexity required. [ADV\_INT.1](#adv_int.1) places requirements for well-structured internals on only selected parts of the TSF. This component is not included in an EAL because this component is viewed for use in special circumstances (e.g. the sponsor has a specific concern regarding a cryptographic module, which is isolated from the rest of the TSF) and would not be widely applicable.
2. At the next level, the requirements for well-structured internals are placed on the entire TSF. Finally, minimization of complexity is introduced in the highest component.

Application notes

1. These requirements, when applied to the internal structure of the TSF, typically result in improvements that aid both the developer and the evaluator in understanding the TSF, and also provide the basis for designing and evaluating test suites. Further, improving understandability of the TSF should assist the developer in simplifying its maintainability.
2. The requirements in this family are presented at a fairly abstract level. The wide variety of TOEs makes it impossible to codify anything more specific than “well-structured” or “minimum complexity”. Judgements on structure and complexity are expected to be derived from the specific technologies used in the TOE. For example, software is likely to be considered well-structured if it exhibits the characteristics cited in the software engineering disciplines. The components within this family call for identifying the standards for measuring the characteristic of being well-structured and not overly-complex.

### Well-structured subset of TSF internals (ADV\_INT.1)

Dependencies: [ADV\_IMP.1](#adv_imp.1), [ADV\_TDS.3](#adv_tds.3), [ALC\_TAT.1](#alc_tat.1)

Developer action elements

ADV\_INT.1.1D The developer shall design and implement [assignment: subset of the TSF] such that it has well-structured internals.

ADV\_INT.1.2D The developer shall provide an internals description and justification.

Content and presentation elements

ADV\_INT.1.1C The justification shall explain the characteristics used to judge the meaning of “well-structured”.

ADV\_INT.1.2C The TSF internals description shall demonstrate that the assigned subset of the TSF is well-structured.

Evaluator action elements

ADV\_INT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_INT.1.2E The evaluator shall perform an internals analysis on the assigned subset of the TSF.

### Well-structured internals (ADV\_INT.2)

Dependencies: [ADV\_IMP.1](#adv_imp.1), [ADV\_TDS.3](#adv_tds.3), [ALC\_TAT.1](#alc_tat.1)

Developer action elements

ADV\_INT.2.1D The developer shall design and implement the **entire** **TSF** such that it has well-structured internals.

ADV\_INT.2.2D The developer shall provide an internals description and justification.

Content and presentation elements

ADV\_INT.2.1C The justification shall **describe** the characteristics used to judge the meaning of “well-structured”.

ADV\_INT.2.2C The TSF internals description shall demonstrate that the **entire** TSF is well-structured.

Evaluator action elements

ADV\_INT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_INT.2.2E The evaluator shall perform an internals analysis on the TSF.

### Minimally complex internals (ADV\_INT.3)

Dependencies: [ADV\_IMP.1](#adv_imp.1), [ADV\_TDS.3](#adv_tds.3), [ALC\_TAT.1](#alc_tat.1)

Developer action elements

ADV\_INT.3.1D The developer shall design and implement the entire TSF such that it has well-structured internals.

ADV\_INT.3.2D The developer shall provide an internals description and justification.

Content and presentation elements

ADV\_INT.3.1C The justification shall describe the characteristics used to judge the meaning of “well-structured” **and** **“complex”.**

ADV\_INT.3.2C The TSF internals description shall demonstrate that the entire TSF is well-structured **and** **is** **not** **overly** **complex.**

Evaluator action elements

ADV\_INT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_INT.3.2E The evaluator shall perform an internals analysis on the **entire** TSF.

## Security policy modelling (ADV\_SPM)

Objectives

1. It is the objective of this family to provide additional assurance through the development of a formal representation of the TSF and its properties, as defined by the SFRs and the security objectives of the ST, further referred to as the formal model and the formal properties, respectively. It is expected to establish by means of a formal proof that these formal properties hold in the formal model and to establish by means of a correspondence rationale that the TOE functional specification preserves the formal properties proven for the formal model. A formal proof or semiformal demonstration of preservation of the formal properties in the formal or semiformal specification is expected if the latter exists ([ADV\_FSP.5](#adv_fsp.5) or [ADV\_FSP.6](#adv_fsp.6), respectively).

Levelling criteria

1. This family contains only one component.

Application notes

1. Inadequacies in a TOE can result either from a failure in understanding the security requirements or from a flawed implementation of those security requirements. Defining the security requirements adequately to ensure their understanding may be problematic because the definition must be sufficiently precise to prevent undesired results or subtle flaws during the implementation of the TOE. Throughout the design, implementation, and review processes, a formal representation of the TSF and its properties may be used as precise design and implementation guidance, thereby providing increased assurance that the SFRs and the security objectives of the ST are satisfied by the TOE. The resulting guidance and the precision of the TSF representation and its properties, as defined by the SFRs and the security objectives of the ST, are significantly improved by defining the formal model and specifying the formal properties using a formal language and providing a formal proof that these formal properties hold in the formal model.
2. 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 on 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 of the ST and the TOE design. 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 TOE functional specification (as defined for [ADV\_FSP](#adv_fsp)). The correspondence rationale consists of a formal proof when mapping to formal aspects of the TOE 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.

### Formal TOE security policy model (ADV\_SPM.1)

Dependencies: [ASE\_OBJ.2](#ase_obj.2), [ASE\_REQ.2](#ase_req.2), [ADV\_FSP.4](#adv_fsp.4)

Developer action elements

ADV\_SPM.1.1D The developer shall provide a formal model for the TSF supported by explanatory text.

ADV\_SPM.1.2D The developer shall provide the set of formal properties for the TOE supported by explanatory text.

ADV\_SPM.1.3D The developer shall provide a formal proof that the model satisfies the formal properties supported by explanatory text.

ADV\_SPM.1.4D The developer shall provide a correspondence rationale between the formal model and the functional specification.

ADV\_SPM.1.5D The developer shall provide a semi-formal demonstration of correspondence between the formal model and any semi-formal functional specification.

ADV\_SPM.1.6D The developer shall provide a formal proof of correspondence between the formal model and any formal functional specification.

Content and presentation elements

ADV\_SPM.1.1C The formal model, properties and proofs shall be defined using a well-founded mathematical theory.

ADV\_SPM.1.2C The explanatory text shall cover the entire formal model, formal properties and proofs, including instructions for reproducing the proofs and the correspondence rationale, and it shall provide a rationale for the modeling and verification choices.

ADV\_SPM.1.3C The formal model shall cover the complete set of SFRs that define the TSF.

ADV\_SPM.1.4C The formal properties shall cover the complete set of security objectives for the TOE.

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.

ADV\_SPM.1.6C The correspondence rationale shall show that the formal properties proven for the formal model hold for the functional specification.

ADV\_SPM.1.7C The semi-formal demonstration of correspondence shall show that the formal properties proven for the formal model hold for any semi-formal functional specification.

ADV\_SPM.1.8C The formal proof of correspondence shall show that the properties proven for the formal model hold for any formal functional specification.

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.

Evaluator action elements

ADV\_SPM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## TOE design (ADV\_TDS)

Objectives

1. The design description of a TOE provides both context for a description of the TSF, and a thorough description of the TSF. As assurance needs increase, the level of detail provided in the description also increases. As the size and complexity of the TSF increase, multiple levels of decomposition are appropriate. The design requirements are intended to provide information (commensurate with the given assurance level) so that a determination can be made that the SFRs are realized.

Levelling criteria

1. The components in this family are levelled on the basis of the amount of information that is required to be presented with respect to the TSF, and on the degree of formalism required of the design description.

Application notes

General

1. The goal of design documentation is to provide sufficient information to determine the TSF boundary, and to describe how the TSF implements the SFRs. The amount and structure of the design documentation will depend on the complexity of the TOE and the number of SFRs; in general, a very complex TOE with a large number of SFRs will require more design documentation than a very simple TOE implementing only a few SFRs. Very complex TOEs will benefit (in terms of the assurance provided) from the production of differing levels of decomposition in describing the design, while very simple TOEs do not require both high-level and low-level descriptions of its implementation.
2. This family uses two levels of decomposition: the subsystem and the module. A module is the most specific description of functionality: it is a description of the implementation. A developer should be able to implement the part of the TOE described by the module with no further design decisions. A subsystem is a description of the design of the TOE; it helps to provide a high-level description of what a portion of the TOE is doing and how. As such, a subsystem may be further divided into lower-level subsystems, or into modules. Very complex TOEs can require several levels of subsystems in order to adequately convey a useful description of how the TOE works. Very simple TOEs, in contrast, might not require a subsystem level of description; the module can clearly describe how the TOE works.
3. The general approach adopted for design documentation is that, as the level of assurance increases, the emphasis of description shifts from the general (subsystem level) to more (module level) detail. In cases where a module-level of abstraction is appropriate because the TOE is simple enough to be described at the module level, yet the level of assurance calls for a subsystem level of description, the module-level description alone will suffice. For complex TOEs, however, this is not the case: an enormous amount of (module-level) detail would be incomprehensible without an accompanying subsystem level of description.
4. This approach follows the general paradigm that providing additional detail about the implementation of the TSF will result in greater assurance that the SFRs are implemented correctly and provide information that can be used to demonstrate this in testing [ATE](#ate).
5. In the requirements for this family, the term interface is used as the means of communication (between two subsystems or modules). It describes how the communication is invoked; this is similar to the details of TSFI (see [ADV\_FSP](#adv_fsp)). The term interaction is used to identify the purpose for communication; it identifies why two subsystems or modules are communicating.

Detail about the subsystems and modules

1. The requirements define collections of details about subsystems and modules to be provided:
2. The subsystems and modules are identified with a simple list of what they are.
3. Subsystems and modules may be categorised (either implicitly or explicitly) as “SFR-enforcing”, “SFR-supporting”, or “SFR-non-interfering”; these terms are used the same as they are used in [ADV\_FSP](#adv_fsp).
4. A subsystem’s behaviour is what it does. The behaviour may also be categorised as SFR-enforcing, SFR-supporting, or SFR-non-interfering. The behaviour of the subsystem is never categorised as more SFR-relevant than the category of the subsystem itself. For example, an SFR-enforcing subsystem can have SFR-enforcing behaviour as well as SFR-supporting or SFR-non-interfering behaviour.
5. A behaviour summary of a subsystem is an overview of the actions it performs (e.g. “The TCP subsystem assembles IP datagrams into reliable byte streams”).
6. A behaviour description of a subsystem is an explanation of everything it does. This description should be at a level of detail that one can readily determine whether the behaviour has any relevance to the enforcement of the SFRs.
7. A description of interactions among or between subsystems or modules identifies the reason that subsystems or modules communicate and characterizes the information that is passed. It need not define the information to the same level of detail as an interface specification. For example, it would be sufficient to say “subsystem X requests a block of memory from the memory manager, which responds with the location of the allocated memory.
8. A description of interfaces provides the details of how the interactions among modules are achieved. Rather than describing the reason the modules are communicating or the purpose of their communication (i.e. the description of interactions), the description of interfaces describes the details of how that communication is accomplished, in terms of the structure and contents of the messages, semaphores, internal process communications.
9. The purpose describes how a module provides their functionality. It provides sufficient detail that no further design decisions are needed. The correspondence between the implementation representation that implements the module, and the purpose of the module should be readily apparent.
10. A module is otherwise described in terms of whatever is identified in the element.

Subsystems and modules, and “SFR-enforcing” are all further explained in greater detail in A.4, [ADV\_TDS](#adv_tds): Subsystems and Modules.

### Basic design (ADV\_TDS.1)

Dependencies: [ADV\_FSP.2](#adv_fsp.2)

Developer action elements

ADV\_TDS.1.1D The developer shall provide the design of the TOE.

ADV\_TDS.1.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

Content and presentation elements

ADV\_TDS.1.1C The design shall describe the structure of the TOE in terms of subsystems.

ADV\_TDS.1.2C The design shall identify all subsystems of the TSF.

ADV\_TDS.1.3C The design shall provide the behaviour summary of each SFR-supporting or SFR-non-interfering TSF subsystem.

ADV\_TDS.1.4C The design shall summarize the SFR-enforcing behaviour of the SFR-enforcing subsystems.

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.

ADV\_TDS.1.6C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Evaluator action elements

ADV\_TDS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_TDS.1.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

### Architectural design (ADV\_TDS.2)

Dependencies: [ADV\_FSP.3](#adv_fsp.3)

Developer action elements

ADV\_TDS.2.1D The developer shall provide the design of the TOE.

ADV\_TDS.2.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

Content and presentation elements

ADV\_TDS.2.1C The design shall describe the structure of the TOE in terms of subsystems.

ADV\_TDS.2.2C The design shall identify all subsystems of the TSF.

ADV\_TDS.2.3C The design shall provide the behaviour summary of each SFR non-interfering subsystem of the TSF.

ADV\_TDS.2.4C The design shall **describe** the SFR-enforcing behaviour of the SFR-enforcing subsystems.

ADV\_TDS.2.5C The design shall summarize the **SFR-supporting** **and** **SFR-non-interfering** behaviour of the SFR-enforcing subsystems.

ADV\_TDS.2.6C The design shall summarize the behaviour of the **SFR-supporting** subsystems.

ADV\_TDS.2.7C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV\_TDS.2.8C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Evaluator action elements

ADV\_TDS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_TDS.2.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

### Basic modular design (ADV\_TDS.3)

Dependencies: [ADV\_FSP.4](#adv_fsp.4)

Developer action elements

ADV\_TDS.3.1D The developer shall provide the design of the TOE.

ADV\_TDS.3.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

Content and presentation elements

ADV\_TDS.3.1C The design shall describe the structure of the TOE in terms of subsystems.

ADV\_TDS.3.2C The design shall describe the TSF in terms of modules.

ADV\_TDS.3.3C The design shall identify all subsystems of the TSF.

ADV\_TDS.3.4C The design shall **provide** **a** **description** of **each** **subsystem** **of** the **TSF.**

ADV\_TDS.3.5C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV\_TDS.3.6C The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.

ADV\_TDS.3.7C The design shall describe **each** SFR-enforcing **module** **in** **terms** of **its** **purpose** **and** **relationship** **with** **other** **modules.**

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.

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.**

ADV\_TDS.3.10C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Evaluator action elements

ADV\_TDS.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_TDS.3.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

### Semiformal modular design (ADV\_TDS.4)

Dependencies: [ADV\_FSP.5](#adv_fsp.5)

Developer action elements

ADV\_TDS.4.1D The developer shall provide the design of the TOE.

ADV\_TDS.4.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

Content and presentation elements

ADV\_TDS.4.1C The design shall describe the structure of the TOE in terms of subsystems.

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.**

ADV\_TDS.4.3C The design shall identify all subsystems of the TSF.

ADV\_TDS.4.4C The design shall provide a **semiformal** description of each subsystem of the TSF, **supported** **by** **informal,** **explanatory** **text** **where** **appropriate.**

ADV\_TDS.4.5C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV\_TDS.4.6C The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.

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.

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.

ADV\_TDS.4.9C The design shall describe each SFR-non-interfering module in terms of its purpose and interaction with other modules.

ADV\_TDS.4.10C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Evaluator action elements

ADV\_TDS.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_TDS.4.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

### Complete semiformal modular design (ADV\_TDS.5)

Dependencies: [ADV\_FSP.5](#adv_fsp.5)

Developer action elements

ADV\_TDS.5.1D The developer shall provide the design of the TOE.

ADV\_TDS.5.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

Content and presentation elements

ADV\_TDS.5.1C The design shall describe the structure of the TOE in terms of subsystems.

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.

ADV\_TDS.5.3C The design shall identify all subsystems of the TSF.

ADV\_TDS.5.4C The design shall provide a semiformal description of each subsystem of the TSF, supported by informal, explanatory text where appropriate.

ADV\_TDS.5.5C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV\_TDS.5.6C The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.

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.**

ADV\_TDS.5.8C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Evaluator action elements

ADV\_TDS.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV\_TDS.5.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

### Complete semiformal modular design with formal high-level design presentation (ADV\_TDS.6)

Dependencies: [ADV\_FSP.6](#adv_fsp.6)

Developer action elements

ADV\_TDS.6.1D The developer shall provide the design of the TOE.

ADV\_TDS.6.2D The developer shall provide a mapping from the TSFI of the functional specification to the lowest level of decomposition available in the TOE design.

ADV\_TDS.6.3D The developer shall provide a formal specification of the TSF subsystems.

ADV\_TDS.6.4D The developer shall provide a proof of correspondence between the formal specifications of the TSF subsystems and of the functional specification.

Content and presentation elements

ADV\_TDS.6.1C The design shall describe the structure of the TOE in terms of subsystems.

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.

ADV\_TDS.6.3C The design shall identify all subsystems of the TSF.

ADV\_TDS.6.4C The design shall provide a semiformal description of each subsystem of the TSF, supported by informal, explanatory text where appropriate.

ADV\_TDS.6.5C The design shall provide a description of the interactions among all subsystems of the TSF.

ADV\_TDS.6.6C The design shall provide a mapping from the subsystems of the TSF to the modules of the TSF.

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.

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.

ADV\_TDS.6.9C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

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.

Evaluator action elements

ADV\_TDS.6.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

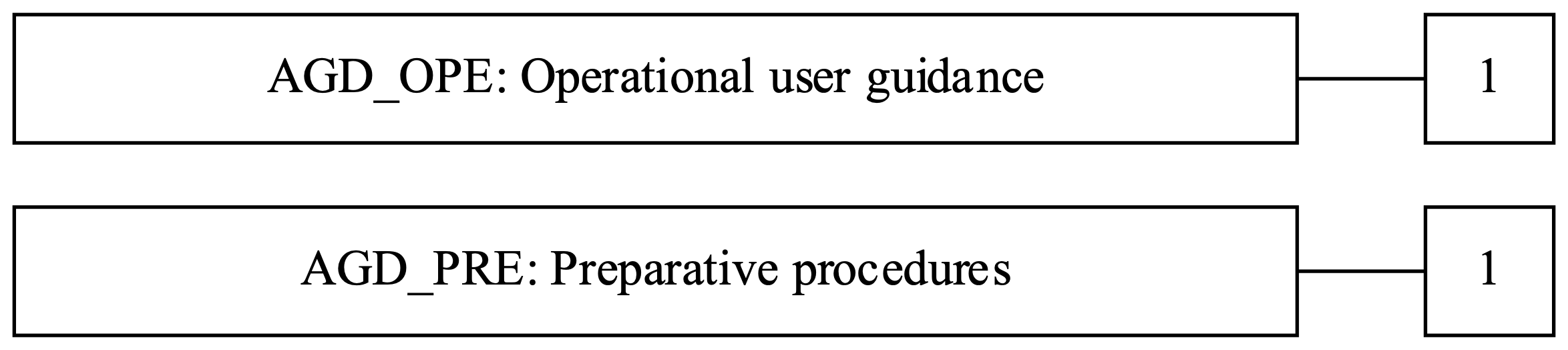
ADV\_TDS.6.2E The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements.

# Class AGD Guidance documents

## Introduction

The guidance documents class provides the requirements for guidance documentation for all user roles. For the secure preparation and operation of the TOE it is necessary to describe all relevant aspects for the secure handling of the TOE. The class also addresses the possibility of unintended incorrect configuration or handling of the TOE.

1. In many cases it can be appropriate to provide guidance in separate documents for preparation and operation of the TOE, or even separate for different user roles as, e.g. end-users, administrators, application programmers using software or hardware interfaces.
2. The guidance documents class is subdivided into two families which are concerned with the preparative user guidance (what has to be done to transform the delivered TOE into its evaluated configuration in the operational environment as described in the ST) and with the operational user guidance (what has to be done during the operation of the TOE in its evaluated configuration).



|  | [ADV\_FSP.1](#adv_fsp.1) |
| --- | --- |
| [**AGD\_OPE.1**](#agd_ope.1) | X |
| [**AGD\_PRE.1**](#agd_pre.1) |  |

## Operational user guidance (AGD\_OPE)

Objectives

1. Operational user guidance refers to written material that is intended to be used by all types of users of the TOE in its evaluated configuration: end-users, persons responsible for maintaining and administering the TOE in a correct manner for maximum security, and by others (e.g. programmers) using the TOE’s external interfaces. Operational user guidance describes the security functionality provided by the TSF, provides instructions and guidelines (including warnings), helps to understand the TSF and includes the security-critical information, and the security-critical actions required, for its secure use. Misleading and unreasonable guidance should be absent from the guidance documentation, and secure procedures for all modes of operation should be addressed. Insecure states should be easy to detect.
2. The operational user guidance provides a measure of confidence that non-malicious users, administrators, application providers and others exercising the external interfaces of the TOE will understand the secure operation of the TOE and will use it as intended. The evaluation of the user guidance includes investigating whether the TOE can be used in a manner that is insecure but that the user of the TOE would reasonably believe to be secure. The objective is to minimize the risk of human or other errors in operation that may deactivate, disable, or fail to activate security functionality, resulting in an undetected insecure state.

Levelling criteria

1. This family contains only one component.

Application notes

1. There may be different user roles or groups that are recognized by the TOE and that can interact with the TSF. These user roles and groups should be taken into consideration by the operational user guidance. They may be roughly grouped into administrators and non-administrative users, or more specifically grouped into persons responsible for receiving, accepting, installing and maintaining the TOE, application programmers, revisors, auditors, daily-management, end-users. Each role can encompass an extensive set of capabilities or can be a single one.
2. The requirement [AGD\_OPE.1.1C](#agd_ope.1.1c) encompasses the aspect that any warnings to the users during operation of a TOE with regard to the security problem definition and the security objectives for the operational environment described in the PP/ST are appropriately covered in the user guidance.
3. The concept of secure values, as employed in [AGD\_OPE.1.3C](#agd_ope.1.3c), has relevance where a user has control over security parameters. Guidance needs to be provided on secure and insecure settings for such parameters.
4. [AGD\_OPE.1.4C](#agd_ope.1.4c) requires that the user guidance describes the appropriate reactions to all security-relevant events. Although many security-relevant events are the result of performing functions, this need not always be the case (e.g. the audit log fills up, an intrusion is detected). Furthermore, a security-relevant event may happen as a result of a specific chain of functions or, conversely, several security-relevant events may be triggered by one function.
5. [AGD\_OPE.1.7C](#agd_ope.1.7c) requires that the user guidance is clear and reasonable. Misleading or unreasonable guidance may result in a user of the TOE believing that the TOE is secure when it is not.
6. An example of misleading guidance would be the description of a single guidance instruction that can be parsed in more than one way, one of which may result in an insecure state.
7. An example of unreasonable guidance would be a recommendation to follow a procedure that is so complicated that it cannot reasonably be expected that users will follow this guidance.

### Operational user guidance (AGD\_OPE.1)

Dependencies: [ADV\_FSP.1](#adv_fsp.1)

Developer action elements

AGD\_OPE.1.1D The developer shall provide operational user guidance.

Content and presentation elements

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.

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.

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.

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.

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.

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.

AGD\_OPE.1.7C The operational user guidance shall be clear and reasonable.

Evaluator action elements

AGD\_OPE.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Preparative procedures (AGD\_PRE)

Objectives

1. Preparative procedures are useful for ensuring that the TOE has been received and installed in a secure manner as intended by the developer. The requirements for preparation call for a secure transition from the delivered TOE to its initial operational environment. This includes investigating whether the TOE can be configured or installed in a manner that is insecure but that the user of the TOE would reasonably believe to be secure.

Levelling criteria

1. This family contains only one component.

Application notes

1. It is recognized that the application of these requirements will vary depending on aspects, e.g. whether the TOE is delivered in an operational state, or whether it has to be installed at the TOE owner’s site.
2. The first process covered by the preparative procedures is the consumer’s secure acceptance of the received TOE in accordance with the developer’s delivery procedures. If the developer has not defined delivery procedures, security of the acceptance has to be ensured otherwise.
3. Installation of the TOE includes transforming its operational environment into a state that conforms to the security objectives for the operational environment provided in the ST.
4. It can also be the case that no installation is necessary, for example a smart card. In this case it may be inappropriate to require and analyse installation procedures.
5. The requirements in this assurance family are presented separately from those in the [AGD\_OPE](#agd_ope) family, due to the infrequent, possibly one-time use of the preparative procedures.

### Preparative procedures (AGD\_PRE.1)

Developer action elements

AGD\_PRE.1.1D The developer shall provide the TOE including its preparative procedures.

Content and presentation elements

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.

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.

Evaluator action elements

AGD\_PRE.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD\_PRE.1.2E The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

# Class ALC Life-cycle support

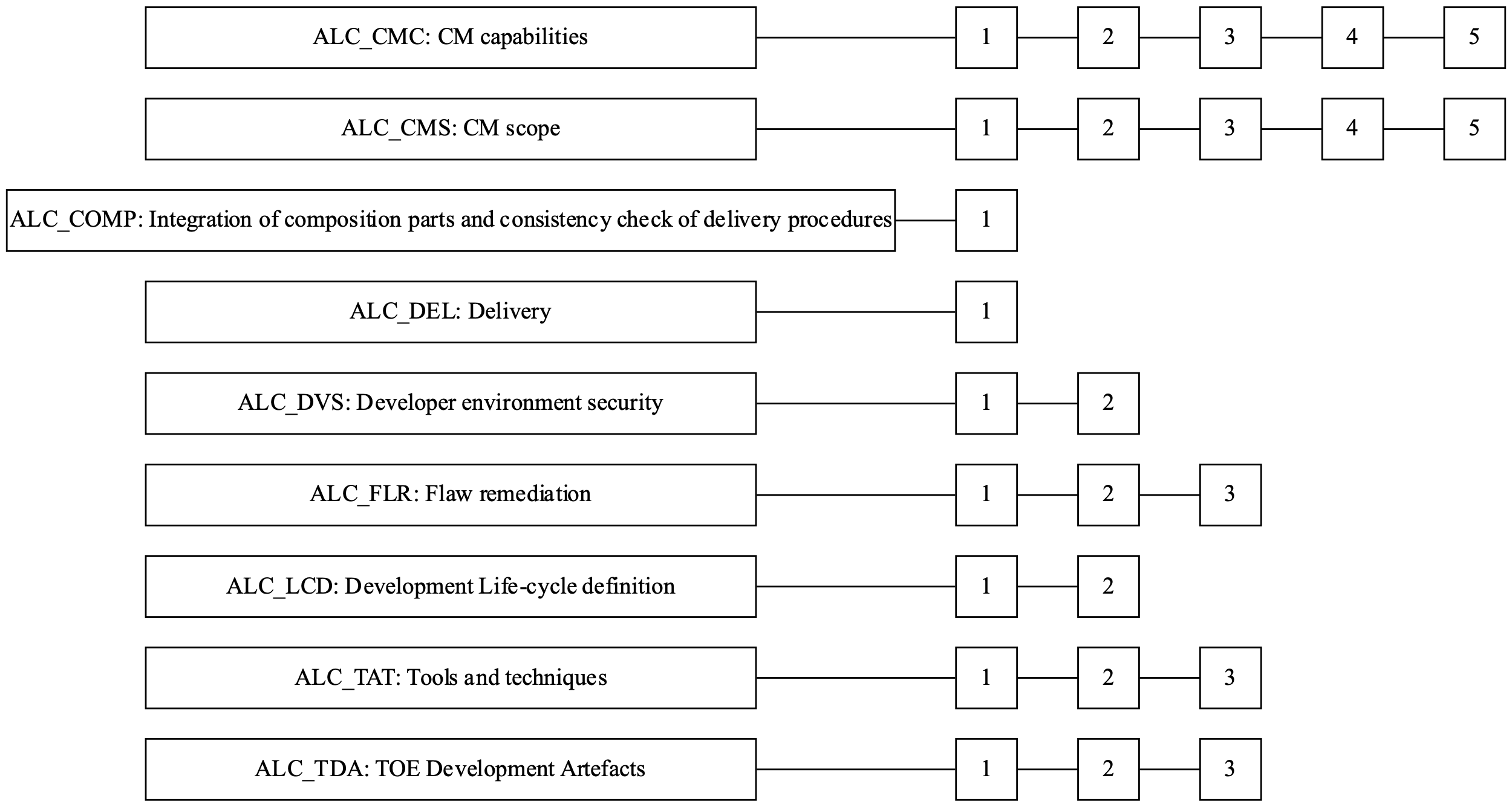
## Introduction

Life-cycle support is an aspect of establishing appropriate security controls in the development, production, delivery and maintenance of the TOE. Confidence in the correspondence between the TOE security requirements and the TOE is greater if security analysis and the production of the evidence are done on a regular basis as an integral part of the development, production, delivery and maintenance activities.

1. During the life-cycle of the TOE it is distinguished whether the TOE is under the responsibility of the TOE developer or the user rather than whether it is located in the development or the user environment. The point of transition is when the TOE is accepted by the user. User in this context relates to the end-user as well as product- and system integrators.
2. The [ALC](#alc) class consists of nine families:

* [ALC\_LCD](#alc_lcd) provides requirements for the developer’s description of the life-cycle model used in the development, production, delivery and maintenance life-cycle of the TOE;
* [ALC\_CMC](#alc_cmc) provides requirements for the management of the configuration items;
* [ALC\_CMS](#alc_cms) requires a minimum set of configuration items to be managed in the defined way;
* [ALC\_DVS](#alc_dvs) is concerned with the developer’s physical, logical, procedural, personnel, and other security controls;
* [ALC\_TAT](#alc_tat) provides requirements for the development tools and implementation standards used by the developer;
* [ALC\_FLR](#alc_flr) provides requirements for the handling of security flaws;
* [ALC\_DEL](#alc_del) provides requirements for the procedures used for the delivery of the TOE to the downstream user. processes occurring during the development of the TOE are denoted rather as transfers, and are handled in the context of integration and acceptance procedures in other families of this class;
* [ALC\_TDA](#alc_tda) is concerned with the generation of certain artefacts during the development process;
* [ALC\_COMP](#alc_comp) is concerned with the integration of composition parts and a consistency check of delivery procedures.

1. Throughout this class, development and related terms (developer, develop) are meant in the more general sense to comprise development and production, whereas production specifically means the process of transforming the implementation representation into the final TOE.



|  | [ADV\_IMP.1](#adv_imp.1) | [ALC\_CMS.1](#alc_cms.1) | [ALC\_CMS.3](#alc_cms.3) | [ALC\_DVS.1](#alc_dvs.1) | [ALC\_DVS.2](#alc_dvs.2) | [ALC\_LCD.1](#alc_lcd.1) | [ALC\_TAT.1](#alc_tat.1) |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [**ALC\_CMC.1**](#alc_cmc.1) |  | X |  |  |  |  |  |
| [**ALC\_CMC.2**](#alc_cmc.2) |  | X |  |  |  |  |  |
| [**ALC\_CMC.3**](#alc_cmc.3) |  | X |  | X |  | X |  |
| [**ALC\_CMC.4**](#alc_cmc.4) |  | X |  | X |  | X |  |
| [**ALC\_CMC.5**](#alc_cmc.5) |  | X |  |  | X | X |  |
| [**ALC\_CMS.1**](#alc_cms.1) |  |  |  |  |  |  |  |
| [**ALC\_CMS.2**](#alc_cms.2) |  |  |  |  |  |  |  |
| [**ALC\_CMS.3**](#alc_cms.3) |  |  |  |  |  |  |  |
| [**ALC\_CMS.4**](#alc_cms.4) |  |  |  |  |  |  |  |
| [**ALC\_CMS.5**](#alc_cms.5) |  |  |  |  |  |  |  |
| [**ALC\_COMP.1**](#alc_comp.1) |  |  |  |  |  |  |  |
| [**ALC\_DEL.1**](#alc_del.1) |  |  |  |  |  |  |  |
| [**ALC\_DVS.1**](#alc_dvs.1) |  |  |  |  |  |  |  |
| [**ALC\_DVS.2**](#alc_dvs.2) |  |  |  |  |  |  |  |
| [**ALC\_FLR.1**](#alc_flr.1) |  |  |  |  |  |  |  |
| [**ALC\_FLR.2**](#alc_flr.2) |  |  |  |  |  |  |  |
| [**ALC\_FLR.3**](#alc_flr.3) |  |  |  |  |  |  |  |
| [**ALC\_LCD.1**](#alc_lcd.1) |  |  |  |  |  |  |  |
| [**ALC\_LCD.2**](#alc_lcd.2) |  |  |  |  |  |  |  |
| [**ALC\_TAT.1**](#alc_tat.1) | X |  |  |  |  |  |  |
| [**ALC\_TAT.2**](#alc_tat.2) | X |  |  |  |  |  |  |
| [**ALC\_TAT.3**](#alc_tat.3) | X |  |  |  |  |  |  |
| [**ALC\_TDA.1**](#alc_tda.1) |  |  |  |  |  |  |  |
| [**ALC\_TDA.2**](#alc_tda.2) |  |  | X |  |  |  |  |
| [**ALC\_TDA.3**](#alc_tda.3) | X |  | X |  |  |  | X |

## CM capabilities (ALC\_CMC)

Objectives

1. Configuration management (CM) techniques, properly defined as part of the development life-cycle model, contribute to the assurance argument that the TOE meets the SFRs. A CM system that is managed and operated correctly will help ensure the integrity of the portions of the TOE that are controlled, by providing a method of tracking any changes to the TOE, and to help ensure that all changes to the TOE are authorized.
2. The objective of this family is to require the TOE developer’s CM system to have certain capabilities. These capabilities are intented to reduce the likelihood that accidental or unauthorised modifications of the configuration items will occur. The CM system should support maintaining the integrity of the TOE throughout the part of the TOE’s life-cycle that is under the control of the developer.
3. The objective of introducing automated CM tools is to increase the effectiveness of the CM system. While both automated and manual CM systems can be bypassed, ignored, or proven insufficient to prevent unauthorised modification, automated systems are less susceptible to human error or negligence.
4. The objectives of this family include the following:
5. ensuring that the TOE is identifiable and complete before it is sent to the downstream user;
6. ensuring that no configuration items are missed during evaluation;
7. preventing unauthorised modification, addition, or deletion of TOE configuration items.

Levelling criteria

1. The components in this family are levelled on the basis of the CM system capabilities, the scope of the CM documentation and the evidence provided by the developer.

Application notes

1. In the case where the TOE is a subset of a product, the requirements of this family apply only to the TOE configuration items, not to the product as a whole.
2. For developer organizations that specify more than one CM application or include different instances of a CM application within the scope of the TOEs design, development, production and maintenance, it is required to document all of them. For evaluation purposes, the set of CM applications should be regarded as parts of an overall CM system, applicable to the TOE, which is addressed in the criteria.
3. The overall CM system should address any aspects of integration between component CM applications.
4. Several elements of this family refer to configuration items. These elements identify CM requirements to be imposed on all items identified in the configuration list, but leave the contents of the list to the discretion of the developer. [ALC\_CMS](#alc_cms) can be used to narrow this discretion by identifying specific items that must be included in the configuration list, and hence within the scope of the overall CM system.
5. [ALC\_CMC.2.3C](#alc_cmc.2.3c) introduces a requirement that the CM system uniquely identify all configuration items. This also requires that modifications to configuration items result in a new, unique identifier being assigned to the configuration item.
6. [ALC\_CMC.3.8C](#alc_cmc.3.8c) introduces the requirement that the evidence shall demonstrate that the CM system operates in accordance with the CM plan. Examples of such evidence can be documentation such as screen snapshots or audit trail output from the CM system, or a detailed demonstration of the CM system by the developer. The evaluator is responsible for determining that this evidence is sufficient to show that the CM system operates in accordance with the CM plan.
7. [ALC\_CMC.4.5C](#alc_cmc.4.5c) introduces a requirement that the CM system provide an automated means to support the production of the TOE. This requires that the CM system provide an automated means to assist in determining that the correct configuration items are used in generating the TOE.
8. [ALC\_CMC.5.10C](#alc_cmc.5.10c) introduces a requirement that the CM system provide an automated means to ascertain the changes between the TOE and its preceding version. If no previous version of the TOE exists, the developer still needs to provide an automated means to ascertain the changes between the TOE and a future version of the TOE.

### Labelling of the TOE (ALC\_CMC.1)

Dependencies: [ALC\_CMS.1](#alc_cms.1)

Developer action elements

ALC\_CMC.1.1D The developer shall provide the TOE and a unique reference for the TOE.

Content and presentation elements

ALC\_CMC.1.1C The TOE shall be labelled with its unique reference.

Evaluator action elements

ALC\_CMC.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Use of the CM system (ALC\_CMC.2)

Dependencies: [ALC\_CMS.1](#alc_cms.1)

Developer action elements

ALC\_CMC.2.1D The developer shall provide the TOE and a unique reference for the TOE.

ALC\_CMC.2.2D The developer shall provide the CM documentation.

ALC\_CMC.2.3D The developer shall use a CM system.

Content and presentation elements

ALC\_CMC.2.1C The TOE shall be labelled with its unique reference.

ALC\_CMC.2.2C The CM documentation shall describe the method used to uniquely identify the configuration items.

ALC\_CMC.2.3C The CM system shall uniquely identify all configuration items.

Evaluator action elements

ALC\_CMC.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Authorization controls (ALC\_CMC.3)

Dependencies: [ALC\_CMS.1](#alc_cms.1), [ALC\_DVS.1](#alc_dvs.1), [ALC\_LCD.1](#alc_lcd.1)

Developer action elements

ALC\_CMC.3.1D The developer shall provide the TOE and a unique reference for the TOE.

ALC\_CMC.3.2D The developer shall provide the CM documentation.

ALC\_CMC.3.3D The developer shall use a CM system.

Content and presentation elements

ALC\_CMC.3.1C The TOE shall be labelled with its unique reference.

ALC\_CMC.3.2C The CM documentation shall describe the method used to uniquely identify the configuration items.

ALC\_CMC.3.3C The CM system shall uniquely identify all configuration items.

ALC\_CMC.3.4C The CM system shall provide controls such that only authorized changes are made to the configuration items.

ALC\_CMC.3.5C The CM documentation shall include a CM plan.

ALC\_CMC.3.6C The CM plan shall describe how the CM system is used for the development of the TOE.

ALC\_CMC.3.7C The evidence shall demonstrate that all configuration items are being maintained under the CM system.

ALC\_CMC.3.8C The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.

Evaluator action elements

ALC\_CMC.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Production support, acceptance procedures and automation (ALC\_CMC.4)

Dependencies: [ALC\_CMS.1](#alc_cms.1), [ALC\_DVS.1](#alc_dvs.1), [ALC\_LCD.1](#alc_lcd.1)

Developer action elements

ALC\_CMC.4.1D The developer shall provide the TOE and a unique reference for the TOE.

ALC\_CMC.4.2D The developer shall provide the CM documentation.

ALC\_CMC.4.3D The developer shall use a CM system.

Content and presentation elements

ALC\_CMC.4.1C The TOE shall be labelled with its unique reference.

ALC\_CMC.4.2C The CM documentation shall describe the method or methods used to uniquely identify the configuration items.

ALC\_CMC.4.3C The CM system shall uniquely identify all configuration items.

ALC\_CMC.4.4C The CM system shall provide **automated** controls such that only authorized changes are made to the configuration items.

ALC\_CMC.4.5C The CM **system** shall **support** the **production** **of** the **TOE** **by** **automated** **means.**

ALC\_CMC.4.6C The CM documentation shall include a CM plan.

ALC\_CMC.4.7C The CM plan shall describe how the CM system is used for the development of the TOE.

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.

ALC\_CMC.4.9C The evidence shall demonstrate that all configuration items are being maintained under the CM system.

ALC\_CMC.4.10C The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.

Evaluator action elements

ALC\_CMC.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Advanced support (ALC\_CMC.5)

Dependencies: [ALC\_CMS.1](#alc_cms.1), [ALC\_DVS.2](#alc_dvs.2), [ALC\_LCD.1](#alc_lcd.1)

Developer action elements

ALC\_CMC.5.1D The developer shall provide the TOE and a unique reference for the TOE.

ALC\_CMC.5.2D The developer shall provide the CM documentation.

ALC\_CMC.5.3D The developer shall use a CM system.

Content and presentation elements

ALC\_CMC.5.1C The TOE shall be labelled with its unique reference.

ALC\_CMC.5.2C The CM documentation shall describe the method used to uniquely identify the configuration items.

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.

ALC\_CMC.5.4C The CM system shall uniquely identify all configuration items.

ALC\_CMC.5.5C The CM system shall provide automated controls such that only authorized changes are made to the configuration items.

ALC\_CMC.5.6C The CM system shall support the production of the TOE by automated means.

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.

ALC\_CMC.5.8C The CM system shall identify the configuration items that comprise the TSF.

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.

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.

ALC\_CMC.5.11C The CM system shall be able to identify the version of the implementation representation from which the TOE is generated.

ALC\_CMC.5.12C The CM documentation shall include a CM plan.

ALC\_CMC.5.13C The CM plan shall describe how the CM system is used for the development of the TOE.

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.

ALC\_CMC.5.15C The evidence shall demonstrate that all configuration items are being maintained under the CM system.

ALC\_CMC.5.16C The evidence shall demonstrate that the CM system is being operated in accordance with the CM plan.

Evaluator action elements

ALC\_CMC.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_CMC.5.2E The evaluator shall determine that the application of the production support procedures results in a TOE as provided by the developer for testing activities.

## CM scope (ALC\_CMS)

Objectives

1. The objective of this family is to identify items to be included as configuration items and hence placed under the CM requirements of [ALC\_CMC](#alc_cmc). Applying configuration management to these additional items provides additional assurance that the integrity of TOE is maintained.

Levelling criteria

1. The components in this family are levelled on the basis of which of the following are required to be included as configuration items: the TOE and the evaluation evidence required by the SARs; the parts of the TOE; the implementation representation; security flaws; and development tools and related information.

Application notes

1. While [ALC\_CMS](#alc_cms) mandates a list of configuration items and that each item on this list be under CM, [ALC\_CMC](#alc_cmc) leaves the contents of the configuration list to the discretion of the developer. [ALC\_CMS](#alc_cms) narrows this discretion by identifying items that must be included in the configuration list, and hence come under the CM requirements of [ALC\_CMC](#alc_cmc).

### TOE CM coverage (ALC\_CMS.1)

Developer action elements

ALC\_CMS.1.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements

ALC\_CMS.1.1C The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC\_CMS.1.2C The configuration list shall uniquely identify the configuration items.

Evaluator action elements

ALC\_CMS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Parts of the TOE CM coverage (ALC\_CMS.2)

Developer action elements

ALC\_CMS.2.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements

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.**

ALC\_CMS.2.2C The configuration list shall uniquely identify the configuration items.

ALC\_CMS.2.3C For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

Evaluator action elements

ALC\_CMS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Implementation representation CM coverage (ALC\_CMS.3)

Developer action elements

ALC\_CMS.3.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements

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.**

ALC\_CMS.3.2C The configuration list shall uniquely identify the configuration items.

ALC\_CMS.3.3C For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

Evaluator action elements

ALC\_CMS.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Problem tracking CM coverage (ALC\_CMS.4)

Developer action elements

ALC\_CMS.4.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements

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.**

ALC\_CMS.4.2C The configuration list shall uniquely identify the configuration items.

ALC\_CMS.4.3C For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

Evaluator action elements

ALC\_CMS.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Development tools CM coverage (ALC\_CMS.5)

Developer action elements

ALC\_CMS.5.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements

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.**

ALC\_CMS.5.2C The configuration list shall uniquely identify the configuration items.

ALC\_CMS.5.3C For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.

Evaluator action elements

ALC\_CMS.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Integration of composition parts and consistency check of delivery procedures (ALC\_COMP)

Objectives

1. The aim of this family 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.

Levelling criteria

1. This family contains only one component.

Application notes

1. 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.
2. 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, the configuration list of the base component developer (e.g. provided within his acknowledgement statement).
3. 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.
4. 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.
5. 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, 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).

### Integration of the dependent component into the related base component and Consistency check for delivery and acceptance procedures (ALC\_COMP.1)

Developer action elements

ALC\_COMP.1.1D The developer shall provide components configuration evidence.

Content and presentation elements

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.

ALC\_COMP.1.2C The components configuration evidence shall show that:

1. 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.
2. 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.

Evaluator action elements

ALC\_COMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_COMP.1.2E The evaluator shall confirm that the evidence for delivery compatibility is complete, coherent, and internally consistent.

## Delivery (ALC\_DEL)

Objectives

1. The concern of this family is the secure transfer of the finished TOE from the development environment into the responsibility of the user.
2. The requirements for delivery call for system control and distribution facilities and procedures that detail the controls necessary to provide assurance that the security of the TOE is maintained during distribution of the TOE to the user. For a valid distribution of the TOE, the procedures used for the distribution of the TOE address the implied or identified objectives identified in the PP/ST relating to the security of the TOE during delivery.

Levelling criteria

1. This family contains only one component. An increasing level of protection for the TOE is established by requiring that the delivery procedures are commensurate with the assumed attack potential in the family [AVA\_VAN](#ava_van) specified in the ST.

Application notes

1. Transfers from subcontractors to the developer or between different development sites are not considered here, but in the family [ALC\_DVS](#alc_dvs).
2. The end of the delivery phase is marked by the acceptance of the transfer of the TOE into the responsibility of the downstream user.
3. NOTE: This does not necessarily coincide with the arrival of the TOE at the downstream user’s location.
4. The delivery procedures should consider, if applicable, issues such as:
5. ensuring that the TOE received by the consumer corresponds precisely to the evaluated version of the TOE;
6. avoiding or detecting any tampering with the actual version of the TOE;
7. preventing submission of a counterfeit version of the TOE;
8. avoiding unwanted knowledge of distribution of the TOE to the consumer: there can be cases where potential attackers should not know when and how it is delivered;
9. avoiding or detecting the TOE being intercepted during delivery; and
10. avoiding the TOE being delayed or stopped during distribution.

The delivery procedures should include the recipient’s actions implied by these issues. The consistent description of these implied actions is examined in the [AGD\_PRE](#agd_pre) family, if present.

### Delivery procedures (ALC\_DEL.1)

Developer action elements

ALC\_DEL.1.1D The developer shall document and provide procedures for delivery of the TOE or parts of it to the consumer.

ALC\_DEL.1.2D The developer shall use the delivery procedures.

Content and presentation elements

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.

Evaluator action elements

ALC\_DEL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Developer environment security (ALC\_DVS)

Objectives

1. Development security is concerned with the determination and specification of security controls relating to the developer provided environment.
2. NOTE: Such controls include coverage of security relevant aspects of asset management, human resources security, physical and environmental security, communications and operations management, access control, information systems acquisition, development and maintenance, information security incident management, and business continuity management.

Levelling criteria

1. The components in this family are levelled on the basis of whether justification of the sufficiency of the security controls is required.

Application notes

1. This family deals with controls to remove or reduce threads and security risks existing at the developer’s site.
2. The evaluator should visit the site(s) in order to assess evidence for development security. This may include sites of subcontractors involved in the TOE development and production. Any decision not to visit shall be agreed with the evaluation authority.
3. Although development security deals with the maintenance of the TOE and hence with aspects becoming relevant after the completion of the evaluation, the [ALC\_DVS](#alc_dvs) requirements specify only that the development security controls be in place at the time of evaluation. Furthermore, [ALC\_DVS](#alc_dvs) does not contain any requirements related to the sponsor’s intention to apply the development security controls in the future, after completion of the evaluation.
4. It is recognized that confidentiality may not always be an issue for the protection of the TOE in its development environment. The use of the word “necessary” allows for the selection of appropriate safeguards.

### Identification of security controls (ALC\_DVS.1)

Developer action elements

ALC\_DVS.1.1D The developer shall produce and provide development security documentation.

Content and presentation elements

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.

Evaluator action elements

ALC\_DVS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_DVS.1.2E The evaluator shall confirm that the security controls are being applied.

### Sufficiency of security controls (ALC\_DVS.2)

Developer action elements

ALC\_DVS.2.1D The developer shall produce and provide development security documentation.

Content and presentation elements

ALC\_DVS.2.1C The development security documentation shall describe all the physical, 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.

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.

Evaluator action elements

ALC\_DVS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_DVS.2.2E The evaluator shall confirm that the security controls are being applied.

## Flaw remediation (ALC\_FLR)

Objectives

1. Flaw remediation requires that discovered security flaws be tracked and corrected by the developer. Although future compliance with flaw remediation procedures cannot be determined at the time of the TOE evaluation, it is possible to evaluate the policies and procedures that a developer has in place to track and correct flaws, and to distribute the flaw information and corrections.

Levelling criteria

1. The components in this family are levelled on the basis of the increasing extent in scope of the flaw remediation procedures and the rigour of the flaw remediation policies.

Application notes

1. This family provides assurance that the TOE will be maintained and supported in the future, requiring the TOE developer to track and correct flaws in the TOE. Additionally, requirements are included for the distribution of flaw corrections. However, this family does not impose evaluation requirements beyond the current evaluation.
2. The TOE user is considered to be the focal point in the user organization that is responsible for receiving and implementing fixes to security flaws. This is not necessarily an individual user, but may be an organisational representative who is responsible for the handling of security flaws. The use of the term TOE user recognizes that different organisations have different procedures for handling flaw reporting, which may be done either by an individual user, or by a central administrative body.
3. The flaw remediation procedures should describe the methods for dealing with all types of flaws encountered. These flaws may be reported by the developer, by users of the TOE, or by other parties with familiarity with the TOE. Some flaws may not be reparable immediately. There may be some occasions where a flaw cannot be fixed and other (e.g. procedural) controls must be taken. The documentation provided should cover the procedures for providing the operational sites with fixes and providing information on flaws where fixes are delayed (and what to do in the interim) or when fixes are not possible.
4. Changes applied to a TOE after its release render it unevaluated; although some information from the original evaluation may still apply. The phrase “release of the TOE” used in this family therefore refers to a version of a product that is a release of a certified TOE, to which changes have been applied.

### Basic flaw remediation (ALC\_FLR.1)

Developer action elements

ALC\_FLR.1.1D The developer shall document and provide flaw remediation procedures addressed to TOE developers.

Content and presentation elements

ALC\_FLR.1.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC\_FLR.1.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC\_FLR.1.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

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.

Evaluator action elements

ALC\_FLR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Flaw reporting procedures (ALC\_FLR.2)

Developer action elements

ALC\_FLR.2.1D The developer shall document and provide flaw remediation procedures addressed to TOE developers.

ALC\_FLR.2.2D The developer shall establish a procedure for accepting and acting upon all reports of security flaws and requests for corrections to those flaws.

ALC\_FLR.2.3D The developer shall provide flaw remediation guidance addressed to TOE users.

Content and presentation elements

ALC\_FLR.2.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC\_FLR.2.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC\_FLR.2.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

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.

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.

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.

ALC\_FLR.2.7C The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.

ALC\_FLR.2.8C The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.

Evaluator action elements

ALC\_FLR.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Systematic flaw remediation (ALC\_FLR.3)

Developer action elements

ALC\_FLR.3.1D The developer shall document and provide flaw remediation procedures addressed to TOE developers.

ALC\_FLR.3.2D The developer shall establish a procedure for accepting and acting upon all reports of security flaws and requests for corrections to those flaws.

ALC\_FLR.3.3D The developer shall provide flaw remediation guidance addressed to TOE users.

Content and presentation elements

ALC\_FLR.3.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC\_FLR.3.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC\_FLR.3.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

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.

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.

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.

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.

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.

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.

ALC\_FLR.3.10C The flaw remediation guidance shall describe a means by which TOE users may register with the developer, to be eligible to receive security flaw reports and corrections.

ALC\_FLR.3.11C The flaw remediation guidance shall identify the specific points of contact for all reports and enquiries about security issues involving the TOE.

Evaluator action elements

ALC\_FLR.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Development Life-cycle definition (ALC\_LCD)

Objectives

1. Poorly defined or uncontrolled processes applied during the development, production and maintenance of the TOE can result in a TOE that does not meet all of its security objectives. Therefore, it is important that well defined and controlled processes be established as early as possible in the TOE’s life-cycle.
2. Defining and implementing such processes does not guarantee that the TOE meets all of its SFRs. It is possible that the processes will be insufficient or inadequate.
3. Adopting a life-cycle model, or models that meets the needs of the developer’s organization will improve the likelihood that the development, production and maintenance processes applied to TOE support the correct design and implementation of a TOE that meets the specified SFRs.
4. The determination of appropriate process controls in order to support process improvement is a long-established best practice.

Levelling criteria

1. The components in this family are levelled on the basis of increasing requirements for measurability of the life-cycle model, and for compliance with that model.

Application notes

1. A life-cycle model encompasses the procedures, tools and techniques used to develop and maintain the TOE. Aspects of the process that may be covered by such a model include design methods, review procedures, project management controls, change control procedures, test methods and acceptance procedures. An effective life-cycle model will address these aspects of the development and maintenance process within an overall management structure that assigns responsibilities and monitors progress.
2. There are different types of acceptance situations that are dealt with at different locations in the criteria:

* acceptance of parts delivered by subcontractors (“integration”) should be treated in this family,
* Life-cycle definition ([ALC\_LCD](#alc_lcd)),
* acceptance subsequent to internal transportations in Development security ([ALC\_DVS](#alc_dvs)),
* acceptance of parts into the CM system in ([ALC\_CMC](#alc_cmc)), and
* acceptance of the delivered TOE by the consumer in ([ALC\_DEL](#alc_del)).

1. The first three types may overlap.
2. Although life-cycle definition deals with the maintenance of the TOE and hence with aspects becoming relevant after the completion of the evaluation, its evaluation adds assurance through an analysis of the life-cycle information for the TOE provided at the time of the evaluation.
3. A life-cycle model provides for the necessary control over the development and maintenance of the TOE, if the model enables sufficient minimization of the danger that the TOE will not meet its security requirement.
4. A measurable life-cycle model is a model using some quantitative valuation (arithmetic parameters and/or metrics) of the managed product in order to measure development properties of the product. Typical metrics are source code complexity metrics, defect density (errors per size of code) or mean time to failure. For the security evaluation all those metrics are of relevance, which are used to increase quality by decreasing the probability of faults and thereby in turn increasing assurance in the security of the TOE.
5. One should take into account that there exist standardised life-cycle models on the one hand (like the waterfall model) and standardised metrics on the other hand (like error density), which may be combined. The ISO/IEC 15408 series does not require the life-cycle to follow exactly one standard defining both aspects.

### Developer defined life-cycle processes (ALC\_LCD.1)

Developer action elements

ALC\_LCD.1.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.

ALC\_LCD.1.2D The developer shall provide life-cycle definition documentation.

Content and presentation elements

ALC\_LCD.1.1C The life-cycle definition documentation shall describe the processes used to develop and maintain the TOE.

ALC\_LCD.1.2C The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.

Evaluator action elements

ALC\_LCD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Measurable life-cycle model (ALC\_LCD.2)

Developer action elements

ALC\_LCD.2.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE **that** **is** **based** **on** **a** **measurable** **life-cycle** **model.**

ALC\_LCD.2.2D The developer shall provide life-cycle definition documentation.

ALC\_LCD.2.3D The developer shall measure the TOE development using the measurable life-cycle model.

ALC\_LCD.2.4D The developer shall provide life-cycle output documentation.

Content and presentation elements

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.**

ALC\_LCD.2.2C The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.

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.

Evaluator action elements

ALC\_LCD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_LCD.2.2E The evaluator shall confirm that the measurements of the TOE development processes and security relevant properties of the TOE support improvements in the development processes and/or the TOE itself.

## Tools and techniques (ALC\_TAT)

Objectives

1. Tools and techniques is an aspect of selecting tools that are used to develop, analyse and implement the TOE. It includes requirements to prevent ill-defined, inconsistent or incorrect development tools from being used to develop the TOE. This includes, but is not limited to, programming languages, documentation, implementation standards, and other parts of the TOE such as supporting runtime libraries.

Levelling criteria

1. The components in this family are levelled on the basis of increasing requirements on the description and scope of the implementation standards and the documentation of implementation-dependent options.

Application notes

1. There is a requirement for well-defined development tools. These are tools that are clearly and completely described. For example, programming languages and computer aided design (CAD) systems that are based on a standard published by standards bodies are considered to be well-defined. Self-made tools would need further investigation to clarify whether they are well-defined.
2. The requirement in [ALC\_TAT.1.2C](#alc_tat.1.2c) is especially applicable to programming languages so as to ensure that all statements in the source code have an unambiguous meaning.
3. In [ALC\_TAT.2](#alc_tat.2) and [ALC\_TAT.3](#alc_tat.3), implementation guidelines may be accepted as an implementation standard if they have been approved by some group of experts (e.g. academic experts, standards bodies). Implementation standards are normally public, well accepted and common practise in a specific industry, but developer-specific implementation guidelines may also be accepted as a standard; the emphasis is on the expertise.
4. Tools and techniques distinguishes between the implementation standards applied by the developer ([ALC\_TAT.2.3D](#alc_tat.2.3d)) and the implementation standards for “all parts of the TOE” ([ALC\_TAT.3.3D](#alc_tat.3.3d)) which include third party software, hardware, or firmware. The configuration list introduced in [ALC\_CMS](#alc_cms) requires that for each TSF relevant configuration item to indicate if it has been generated by the TOE developer or by third party developers.

### Well-defined development tools (ALC\_TAT.1)

Dependencies: [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ALC\_TAT.1.1D The developer shall provide the documentation identifying each development tool being used for the TOE.

ALC\_TAT.1.2D The developer shall document and provide the selected implementation-dependent options of each development tool.

Content and presentation elements

ALC\_TAT.1.1C Each development tool used for implementation shall be well-defined.

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.

ALC\_TAT.1.3C The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.

Evaluator action elements

ALC\_TAT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Compliance with implementation standards (ALC\_TAT.2)

Dependencies: [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ALC\_TAT.2.1D The developer shall provide the documentation identifying each development tool being used for the TOE.

ALC\_TAT.2.2D The developer shall document and provide the selected implementation-dependent options of each development tool.

ALC\_TAT.2.3D The developer shall describe and provide the implementation standards that are being applied by the developer.

Content and presentation elements

ALC\_TAT.2.1C Each development tool used for implementation shall be well-defined.

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.

ALC\_TAT.2.3C The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.

Evaluator action elements

ALC\_TAT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_TAT.2.2E The evaluator shall confirm that the implementation standards have been applied.

### Compliance with implementation standards - all parts (ALC\_TAT.3)

Dependencies: [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ALC\_TAT.3.1D The developer shall provide the documentation identifying each development tool being used for the TOE.

ALC\_TAT.3.2D The developer shall document and provide the selected implementation-dependent options of each development tool.

ALC\_TAT.3.3D The developer shall describe and provide the implementation standards that are being applied by the developer **and** **by** **any** **third-party** **providers** **for** **all** **parts** **of** **the** **TOE.**

Content and presentation elements

ALC\_TAT.3.1C Each development tool used for implementation shall be well-defined.

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.

ALC\_TAT.3.3C The documentation of each development tool shall unambiguously define the meaning of all implementation-dependent options.

Evaluator action elements

ALC\_TAT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_TAT.3.2E The evaluator shall confirm that the implementation standards have been applied.

## TOE Development Artefacts (ALC\_TDA)

Objectives

1. This family aims to add trust to the development process or a development. It focuses on the generation of certain artefacts in the development process. These artefacts are used at a later point in time to assess the degree to which the development process is trustable. This trust is realized through the validation of the generated artefacts for confirming them as sufficient evidence for trustable development.
2. This family introduces developer practices within the development process to generate the required artefacts for realizing trustable development. If a requirement in this family does not explicitly specify the use of automation to generate the required artefacts, the developer is free to undertake the corresponding practice manually, or to use some integrated automation in the development process, or to use a hybrid method of both. It is expected that the degree of trust in the development process is proportional to the degree of automation adoption to implement the corresponding practice in the development process.
3. This family also has a relationship with the [ALC\_TAT](#alc_tat) family. As [ALC\_TAT](#alc_tat) focuses on the tools and techniques aspect for developing, analysing, and implementing the TOE, it provides the necessary context when describing the practices of this family being introduced into the development process.

Levelling criteria

1. The components in this family are levelled on the basis of increasing cross-checking for consistency with relevant evidence from components of other families of other security assurance classes.

Application notes

1. The requirements in [ALC\_TDA.1](#alc_tda.1) provide a degree of trust in the developer’s ability to identify the set of implementation representation which actually has been used during the TOE generation time. This degree of trust helps to positively answer the question “is that really the source code for this software” or “is that really the register-transfer level (RTL) design or description for this integrated circuit hardware”” or “is that really the set of implementation representation for this TOE”, which is potentially relevant in an evaluation. Such degree of trust is built on
2. the timing of when the set of implementation representation identifiers is recorded or logged,
3. the integrity and authenticity of the record of implementation representation identifiers, and
4. the traceability of implementation representation identifiers from the TOE.

In the case where some implementation representation elements are also covered in the configuration list due to [ALC\_CMS.3](#alc_cms.3), the requirements in [ALC\_TDA.2](#alc_tda.2) make sure that these implementation representation elements actually are identifiable through the use of the implementation representation identifiers of [ALC\_TDA.1](#alc_tda.1).

1. With the accurate recording or logging of the actual implementation representation being used by the development tools under the scope of [ALC\_TAT](#alc_tat), it provides an additional evidence to convince a third party that a regeneration of the TOE is functionally equivalent to the original TOE.
2. The requirements in [ALC\_TDA.3](#alc_tda.3) provide the developer an opportunity to testify the absence of functional differences between the two possibly visibly different TOEs which have been independently generated from the identical set of implementation representation.

### Uniquely identifying implementation representation (ALC\_TDA.1)

Developer action elements

ALC\_TDA.1.1D The developer shall identify individual elements of the TOE implementation representation to record the list of unique TOE implementation representation identifiers, as the development tool generates the TOE.

ALC\_TDA.1.2D The developer shall use the current date and time to timestamp the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.1.3D The developer shall maintain the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.1.4D The developer shall ensure the authenticity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time, with the maintenance of the (author) origination information.

ALC\_TDA.1.5D The developer shall be able to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.1.6D The developer shall produce and provide documentation describing

1. the developer’s creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
2. the developer’s timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
3. the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
4. 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;
5. 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.

Content and presentation elements

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.

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.

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.

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.

Evaluator action elements

ALC\_TDA.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_TDA.1.2E The evaluator shall confirm that the development tool for generating the TOE is capable to use or reference the implementation representation element names.

ALC\_TDA.1.3E The evaluator shall confirm that the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the creation time of the TOE.

ALC\_TDA.1.4E The evaluator shall confirm that the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the (author) origination information of the TOE.

ALC\_TDA.1.5E 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.

ALC\_TDA.1.6E The evaluator shall confirm the developer’s ability to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

### Matching CMS scope of implementation representation (ALC\_TDA.2)

Dependencies: [ALC\_CMS.3](#alc_cms.3)

Developer action elements

ALC\_TDA.2.1D The developer shall identify individual elements of the TOE implementation representation to record the list of unique TOE implementation representation identifiers, as the development tool generates the TOE.

ALC\_TDA.2.2D The developer shall use the current date and time to timestamp the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.2.3D The developer shall maintain the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.2.4D The developer shall ensure the authenticity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time, with the maintenance of the (author) origination information.

ALC\_TDA.2.5D The developer shall be able to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.2.6D The developer shall produce and provide documentation describing

1. the developer’s creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
2. the developer’s timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
3. the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
4. 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;
5. 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.

ALC\_TDA.2.7D The developer shall provide evidence that the elements of implementation representation under the configuration scope of ALC\_CMS.3 are identified by the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

Content and presentation elements

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.

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.

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.

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.

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.

Evaluator action elements

ALC\_TDA.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_TDA.2.2E The evaluator shall confirm that the development tool for generating the TOE is capable to use or reference the implementation representation element names.

ALC\_TDA.2.3E The evaluator shall confirm that the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the creation time of the TOE.

ALC\_TDA.2.4E The evaluator shall confirm that the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the (author) origination information of the TOE.

ALC\_TDA.2.5E 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.

ALC\_TDA.2.6E The evaluator shall confirm the developer’s ability to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.2.7E The evaluator shall confirm that the list of identifiers of the elements of implementation representation under the configuration scope of ALC\_CMS.3 matches with the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

### Regenerate TOE with well-defined development tools (ALC\_TDA.3)

Dependencies: [ALC\_CMS.3](#alc_cms.3), [ALC\_TAT.1](#alc_tat.1), [ADV\_IMP.1](#adv_imp.1)

Developer action elements

ALC\_TDA.3.1D The developer shall identify individual elements of the TOE implementation representation to record the list of unique TOE implementation representation identifiers, as the development tool generates the TOE.

ALC\_TDA.3.2D The developer shall use the current date and time to timestamp the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.3.3D The developer shall maintain the integrity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.3.4D The developer shall ensure the authenticity of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time, with the maintenance of the (author) origination information.

ALC\_TDA.3.5D The developer shall be able to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.3.6D The developer shall produce and provide documentation describing

1. the developer’s creation of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
2. the developer’s timestamp being applied to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
3. the maintenance of the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time;
4. 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;
5. 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.

ALC\_TDA.3.7D The developer shall provide evidence that the elements of implementation representation under the configuration scope of ALC\_CMS.3 are identified by the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.3.8D After applying the development tools to another copy of the TOE implementation representation according to the list of unique TOE implementation representation identifiers to regenerate a TOE copy, the developer shall explain the functional differences, if any, between the TOE copy and the original TOE.

ALC\_TDA.3.9D The developer shall produce and provide documentation explaining the functional differences, if any, between the regenerated TOE copy and the original TOE.

Content and presentation elements

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.

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.

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.

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.

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.

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

Evaluator action elements

ALC\_TDA.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC\_TDA.3.2E The evaluator shall confirm that the development tool for generating the TOE is capable to use or reference the implementation representation element names.

ALC\_TDA.3.3E The evaluator shall confirm that the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the creation time of the TOE.

ALC\_TDA.3.4E The evaluator shall confirm that the (author) origination information of the list of unique TOE implementation representation identifiers as recorded during the TOE generation time is consistent with the (author) origination information of the TOE.

ALC\_TDA.3.5E 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.

ALC\_TDA.3.6E The evaluator shall confirm the developer’s ability to trace from the TOE to the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

ALC\_TDA.3.7E The evaluator shall confirm that the list of identifiers of the elements of implementation representation under the configuration scope of ALC\_CMS.3 matches with the list of unique TOE implementation representation identifiers as recorded during the TOE generation time.

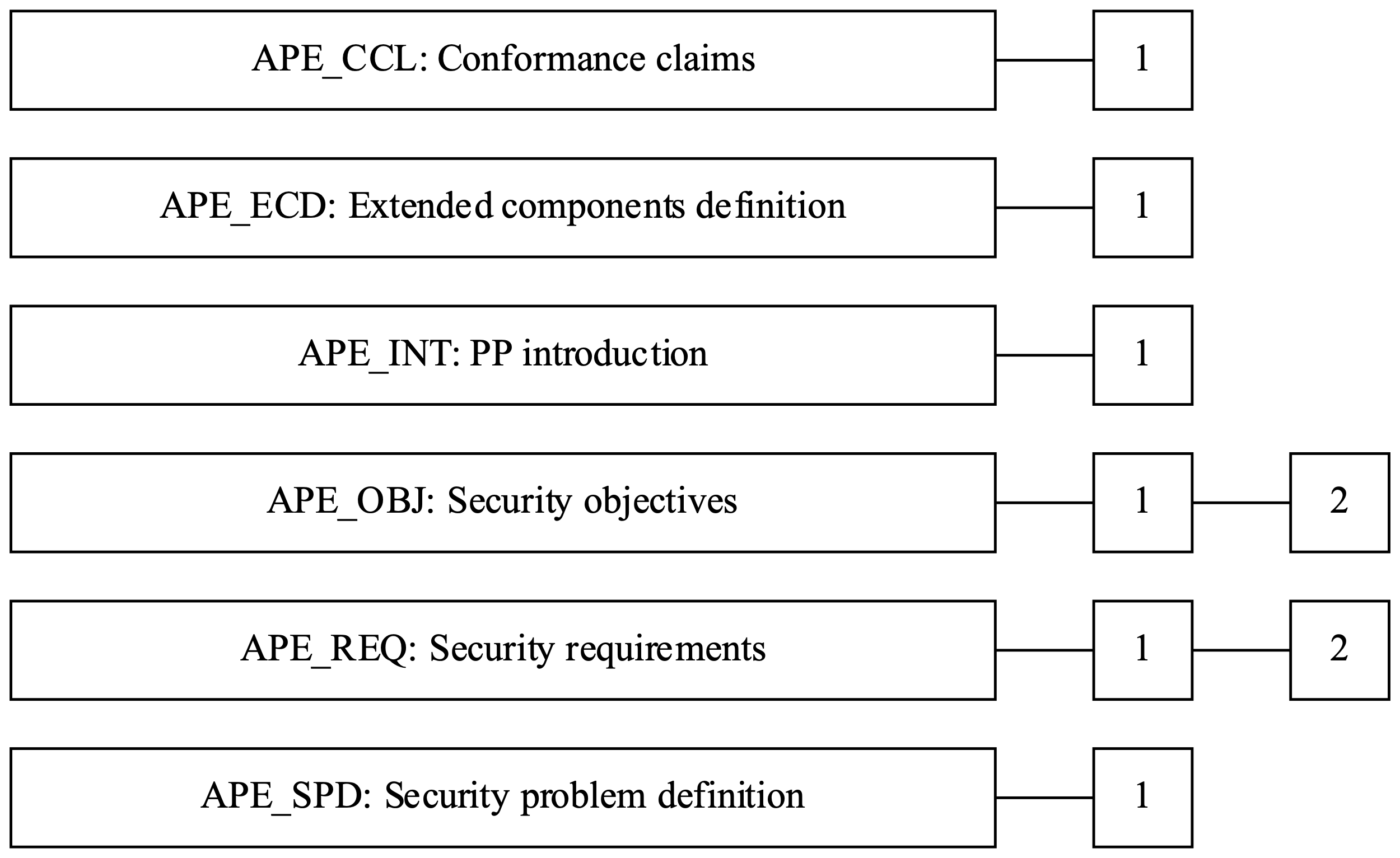
ALC\_TDA.3.8E The evaluator shall check that the developer’s explanation of the functional differences, if any, between the regenerated TOE copy and the original TOE takes into account all visible differences, if any, between the regenerated TOE copy and the original TOE.

# Class APE Protection Profile (PP) evaluation

## Introduction

Evaluating a PP is required to demonstrate that the PP is sound and internally consistent, and, if the PP is based on one or more other PPs or on packages, that the PP is a correct instantiation of these PPs and packages. These properties are necessary for the PP to be suitable for use as the basis for writing an ST or another PP.

1. Clause 7 should be used in conjunction with ISO/IEC 15408-1:2022, Annexes B and D, as these annexes clarify the concepts here and provide many examples.



|  | [APE\_ECD.1](#ape_ecd.1) | [APE\_INT.1](#ape_int.1) | [APE\_OBJ.1](#ape_obj.1) | [APE\_OBJ.2](#ape_obj.2) | [APE\_REQ.1](#ape_req.1) | [APE\_SPD.1](#ape_spd.1) |
| --- | --- | --- | --- | --- | --- | --- |
| [**APE\_CCL.1**](#ape_ccl.1) | X | X | - |  | X |  |
| [**APE\_ECD.1**](#ape_ecd.1) |  |  |  |  |  |  |
| [**APE\_INT.1**](#ape_int.1) |  |  |  |  |  |  |
| [**APE\_OBJ.1**](#ape_obj.1) |  |  |  |  |  |  |
| [**APE\_OBJ.2**](#ape_obj.2) |  |  |  |  |  | X |
| [**APE\_REQ.1**](#ape_req.1) | X |  | X |  |  |  |
| [**APE\_REQ.2**](#ape_req.2) | X |  |  | X |  | - |
| [**APE\_SPD.1**](#ape_spd.1) |  |  |  |  |  |  |

## Conformance claims (APE\_CCL)

Objectives

1. The objective of this family is to determine the validity of the conformance claim. In addition, this family specifies how STs and other PPs are to claim conformance with the PP.

### Conformance claims (APE\_CCL.1)

Dependencies: [APE\_INT.1](#ape_int.1), [APE\_ECD.1](#ape_ecd.1), [APE\_REQ.1](#ape_req.1)

Developer action elements

APE\_CCL.1.1D The developer shall provide a conformance claim.

APE\_CCL.1.2D The developer shall provide a conformance claim rationale.

APE\_CCL.1.3D The developer shall provide a conformance statement.

Content and presentation elements

APE\_CCL.1.1C The conformance claim shall identify the ISO/IEC 15408 edition to which the PP claims conformance.

APE\_CCL.1.2C The conformance claim shall describe the conformance of the PP to ISO/IEC 15408-2 as either ISO/IEC 15408-2 conformant or ISO/IEC 15408-2 extended.

APE\_CCL.1.3C The conformance claim shall describe the conformance of the PP as either “ISO/IEC 15408-3 conformant” or “ISO/IEC 15408-3 extended”.

APE\_CCL.1.4C The conformance claim shall be consistent with the extended components definition.

APE\_CCL.1.5C The conformance claim shall identify all PPs and packages to which the PP claims conformance.

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.

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.

APE\_CCL.1.8C The conformance claim shall describe any conformance of the PP to another PP as PP Conformant.

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.

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.

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.

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.

APE\_CCL.1.13C The conformance statement shall describe the conformance required of any PPs/STs to the PP as one of exact, strict, or demonstrable conformance.

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.

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.

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:

1. any Evaluation methods and Evaluation activities that are specified for the PP under evaluation;
2. any Evaluation methods and Evaluation activities specified in conformance statements of PPs to which conformance is being claimed by the PP under evaluation;
3. 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.

Evaluator action elements

APE\_CCL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Extended components definition (APE\_ECD)

Objectives

1. Extended security requirements are requirements that are not based on components from ISO/IEC 15408-2 or this document, but which are based on extended components: components defined by the PP author.
2. Evaluation of the definition of extended components is necessary to determine that they are clear and unambiguous, and that they are necessary, i.e. they may not be clearly expressed using existing ISO/IEC 15408-2 or this document components.

### Extended components definition (APE\_ECD.1)

Developer action elements

APE\_ECD.1.1D The developer shall provide a statement of security requirements.

APE\_ECD.1.2D The developer shall provide an extended components definition.

Content and presentation elements

APE\_ECD.1.1C The statement of security requirements shall identify all extended security requirements.

APE\_ECD.1.2C The extended components definition shall define an extended component for each extended security requirement.

APE\_ECD.1.3C The extended components definition shall describe how each extended component is related to the existing ISO/IEC 15408 series components, families, and classes.

APE\_ECD.1.4C The extended components definition shall use the existing ISO/IEC 15408 series components, families, classes, and methodology as a model for presentation.

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.

Evaluator action elements

APE\_ECD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

APE\_ECD.1.2E The evaluator shall confirm that no extended component may be clearly expressed using existing components.

## PP introduction (APE\_INT)

Objectives

1. The objective of this family is to describe the TOE in a narrative way.
2. Evaluation of the PP introduction is required to demonstrate that the PP is correctly identified, and that the PP reference and TOE overview are consistent with each other.

### PP introduction (APE\_INT.1)

Developer action elements

APE\_INT.1.1D The developer shall provide a PP introduction.

Content and presentation elements

APE\_INT.1.1C The PP introduction shall contain a PP reference and a TOE overview.

APE\_INT.1.2C The PP reference shall uniquely identify the PP.

APE\_INT.1.3C The TOE overview shall summarize the usage and major security features of the TOE.

APE\_INT.1.4C The TOE overview shall identify the TOE type.

APE\_INT.1.5C The TOE overview shall identify any non-TOE hardware/software/firmware available to the TOE.

Evaluator action elements

APE\_INT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Security objectives (APE\_OBJ)

Objectives

1. The security objectives are a concise statement of the intended response to the security problem defined through the [APE\_SPD](#ape_spd) family.
2. Evaluation of the security objectives is required to demonstrate that 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.

Levelling criteria

1. The components in this family are levelled on whether they prescribe only security objectives for the operational environment, or also security objectives for the TOE.

### Security objectives for the operational environment (APE\_OBJ.1)

Developer action elements

APE\_OBJ.1.1D The developer shall provide a statement of security objectives for the operational environment.

APE\_OBJ.1.2D The developer shall provide a security objectives rationale objectives for the operational environment.

Content and presentation elements

APE\_OBJ.1.1C The statement of security objectives shall describe the security objectives for the operational environment.

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.

APE\_OBJ.1.3C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

APE\_OBJ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Security objectives (APE\_OBJ.2)

Dependencies: [APE\_SPD.1](#ape_spd.1)

Developer action elements

APE\_OBJ.2.1D The developer shall provide a statement of security objectives.

APE\_OBJ.2.2D The developer shall provide a security objectives rationale.

Content and presentation elements

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.

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.

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.

APE\_OBJ.2.4C The security objectives rationale shall demonstrate that the security objectives counter all threats.

APE\_OBJ.2.5C The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.

APE\_OBJ.2.6C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

APE\_OBJ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Security requirements (APE\_REQ)

Objectives

1. The SFRs form a clear, unambiguous and well-defined description of the expected security behaviour of the TOE. The SARs form a clear, unambiguous and well-defined description of the expected activities that will be undertaken to gain assurance in the TOE.
2. Evaluation of the security requirements is required to ensure that they are clear, unambiguous and well-defined.

Levelling criteria

1. The components in this family are levelled on whether the SFRs are derived from SPD, or whether the SFRs are derived from security objectives for the TOE.

### Direct rationale PP-Module security requirements (APE\_REQ.1)

Dependencies: [APE\_ECD.1](#ape_ecd.1), [APE\_OBJ.1](#ape_obj.1)

Developer action elements

APE\_REQ.1.1D The developer shall provide a statement of security requirements.

APE\_REQ.1.2D The developer shall provide a security requirements rationale.

Content and presentation elements

APE\_REQ.1.1C The statement of security requirements shall describe the SFRs and the SARs.

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.

APE\_REQ.1.3C The statement of security requirements shall identify all operations on the security requirements.

APE\_REQ.1.4C All operations shall be performed correctly.

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.

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.

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.

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.

APE\_REQ.1.9C The security requirements rationale shall explain why the SARs were chosen.

APE\_REQ.1.10C The statement of security requirements shall be internally consistent.

Evaluator action elements

APE\_REQ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Derived security requirements (APE\_REQ.2)

Dependencies: [APE\_OBJ.2](#ape_obj.2), [APE\_ECD.1](#ape_ecd.1)

Developer action elements

APE\_REQ.2.1D The developer shall provide a statement of security requirements.

APE\_REQ.2.2D The developer shall provide a security requirements rationale.

Content and presentation elements

APE\_REQ.2.1C The statement of security requirements shall describe the SFRs and the SARs.

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.

APE\_REQ.2.3C The statement of security requirements shall identify all operations on the security requirements.

APE\_REQ.2.4C All operations shall be performed correctly.

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.

APE\_REQ.2.6C The security requirements rationale shall trace each SFR back to the **security** **objectives** **for** the **TOE** enforced by that SFR.

APE\_REQ.2.7C The security requirements rationale shall demonstrate that the SFRs **meet** **all** security objectives for the TOE.

APE\_REQ.2.8C The security requirements rationale shall explain why the SARs were chosen.

APE\_REQ.2.9C The statement of security requirements shall be internally consistent.

Evaluator action elements

APE\_REQ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Security problem definition (APE\_SPD)

Objectives

1. This part of the PP defines the security problem to be addressed by the TOE and the operational environment of the TOE.
2. Evaluation of the security problem definition is required to demonstrate that the security problem intended to be addressed by the TOE and its operational environment, is clearly defined.

### Security problem definition (APE\_SPD.1)

Developer action elements

APE\_SPD.1.1D The developer shall provide a security problem definition.

Content and presentation elements

APE\_SPD.1.1C The security problem definition shall describe the threats.

APE\_SPD.1.2C All threats shall be described in terms of a threat agent, an asset, and an adverse action.

APE\_SPD.1.3C The security problem definition shall describe the organizational security policies (OSPs).

APE\_SPD.1.4C The security problem definition shall describe the assumptions about the operational environment of the TOE.

Evaluator action elements

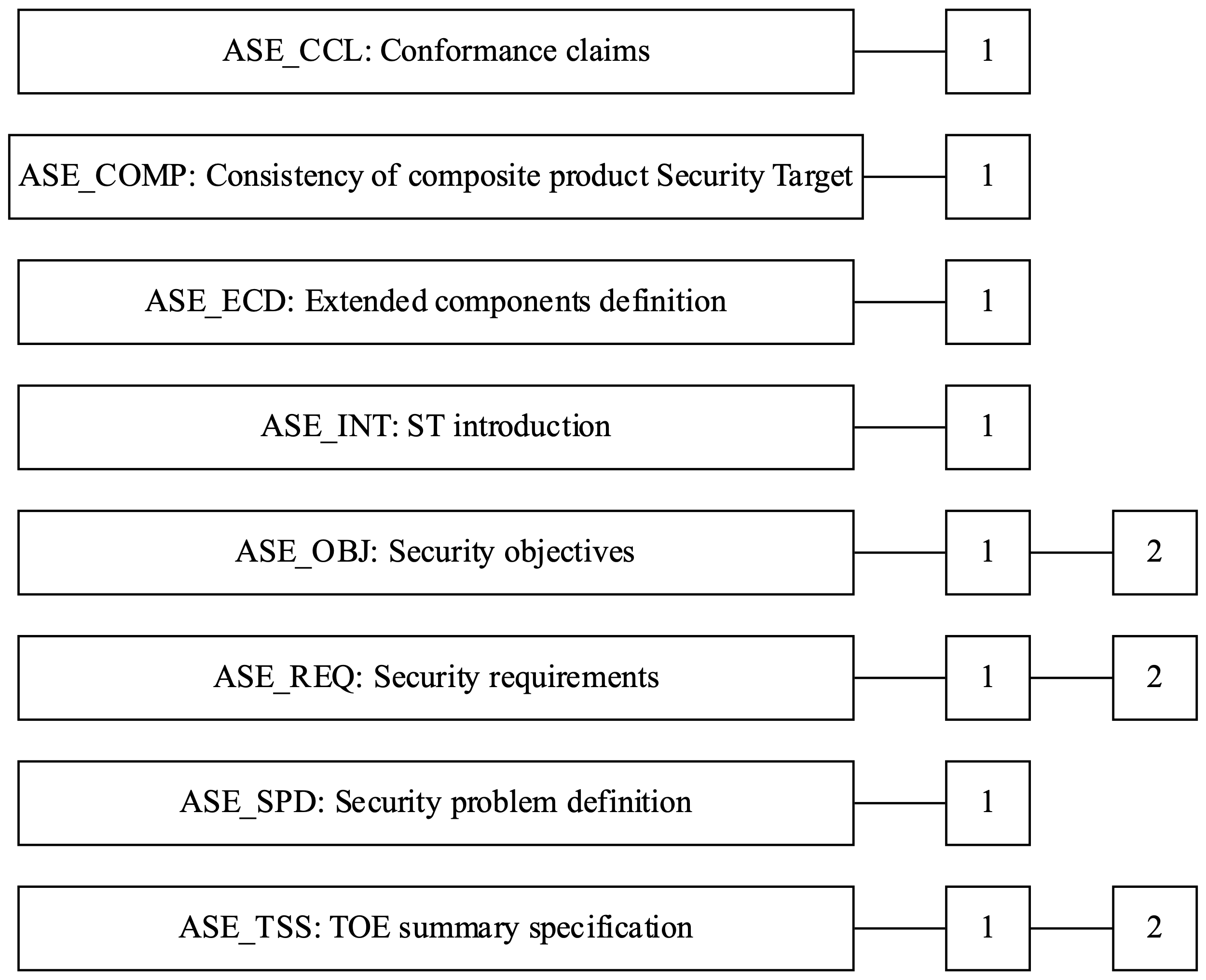
APE\_SPD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

# Class ASE Security Target (ST) evaluation

## Introduction

Evaluating an ST is required to demonstrate that the ST is sound and internally consistent, and, if the ST is based on a PP-Configuration, or one or more PPs or packages, that the ST is a correct instantiation of the PP-Configuration, PPs, and packages. These properties are necessary for the ST to be suitable for use as the basis for a TOE evaluation.

1. Clause 9 should be used in conjunction with ISO/IEC 15408-1:2022, Annexes B, C and D as these annexes clarify the concepts here and provide many examples.



|  | [ADV\_ARC.1](#adv_arc.1) | [ADV\_FSP.1](#adv_fsp.1) | [ASE\_ECD.1](#ase_ecd.1) | [ASE\_INT.1](#ase_int.1) | [ASE\_OBJ.2](#ase_obj.2) | [ASE\_REQ.1](#ase_req.1) | [ASE\_SPD.1](#ase_spd.1) |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [**ASE\_CCL.1**](#ase_ccl.1) |  |  | X | X |  | X |  |
| [**ASE\_COMP.1**](#ase_comp.1) |  |  |  |  |  |  |  |
| [**ASE\_ECD.1**](#ase_ecd.1) |  |  |  |  |  |  |  |
| [**ASE\_INT.1**](#ase_int.1) |  |  |  |  |  |  |  |
| [**ASE\_OBJ.1**](#ase_obj.1) |  |  |  |  |  |  |  |
| [**ASE\_OBJ.2**](#ase_obj.2) |  |  |  |  |  |  | X |
| [**ASE\_REQ.1**](#ase_req.1) |  |  | X |  |  |  |  |
| [**ASE\_REQ.2**](#ase_req.2) |  |  | X |  | X |  | - |
| [**ASE\_SPD.1**](#ase_spd.1) |  |  |  |  |  |  |  |
| [**ASE\_TSS.1**](#ase_tss.1) |  | X | - | X |  | X |  |
| [**ASE\_TSS.2**](#ase_tss.2) | X |  | - | X |  | X |  |

## Conformance claims (ASE\_CCL)

Objectives

1. The objective of this family is to determine the validity of the conformance claim. In addition, this family specifies how STs are to claim conformance with the PP or PP-Configuration.

### Conformance claims (ASE\_CCL.1)

Dependencies: [ASE\_INT.1](#ase_int.1), [ASE\_ECD.1](#ase_ecd.1), [ASE\_REQ.1](#ase_req.1)

Developer action elements

ASE\_CCL.1.1D The developer shall provide a conformance claim.

ASE\_CCL.1.2D The developer shall provide a conformance claim rationale.

Content and presentation elements

ASE\_CCL.1.1C The conformance claim shall identify the edition of ISO/IEC 15408 to which the ST and the TOE claim conformance.

ASE\_CCL.1.2C The conformance claim shall describe the conformance of the ST to ISO/IEC 15408-2 as either ISO/IEC 15408-2 conformant or ISO/IEC 15408-2 extended.

ASE\_CCL.1.3C The conformance claim shall describe the conformance of the ST as either “ISO/IEC 15408-3 conformant” or “ISO/IEC 15408-3 extended”.

ASE\_CCL.1.4C The conformance claim shall be consistent with the extended components definition.

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.

ASE\_CCL.1.6C The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.

ASE\_CCL.1.7C The conformance claim shall describe any conformance of the ST to a PP as PP-Conformant.

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.

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, PPs and any functional packages for which conformance is being claimed.

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, PPs, and any functional package for which conformance is being claimed.

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, PPs, and any functional packages for which conformance is being claimed.

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.

ASE\_CCL.1.13C If the conformance claim identifies a set of Evaluation methods and Evaluation activities derived from ISO/IEC 18045 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.

Evaluator action elements

ASE\_CCL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Consistency of composite product Security Target (ASE\_COMP)

Objectives

1. The aim of this family is to determine whether the ST of the composite product [[1]](#footnote-1) does not contradict the ST of the related base component [[2]](#footnote-2) [[3]](#footnote-3).

Levelling criteria

1. This family contains only one component.

Application notes

1. A ST for the composite product shall be written and evaluated.
2. The composite product evaluator shall examine that the ST of the composite product does not contradict the ST of the related base component. In particular, it means that the composite product evaluator shall examine the composite product ST and the base component ST for any conflicting assumptions, compatibility of security objectives, security requirements and security functionality needed by the dependent component.
3. The composite product evaluation sponsor shall ensure that the ST 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 ST may not be sufficient.
4. These application notes aid the developer to create as well as the evaluator to analyse a composite product ST and describe a general methodology for it.
5. In order to create a composite product ST, the developer should perform the following steps:
6. Step 1: The developer formulates a preliminary ST for the composite product (the composite-ST) using the standard code of practice. The composite-ST can be formulated independently of the ST of the composite product’s related base component (the base-ST), at least as long as there are no formal PP conformance claims.
7. Step 2: The developer determines the overlap between the base-ST and the composite-ST through analysing and comparing their respective TOE Security Functionality (TSF) [[4]](#footnote-4) [[5]](#footnote-5).
8. Step 3: The developer determines under which conditions he can trust in and rely on the base component-TSF being used by the composite-ST without a new examination.
9. Having undertaken these steps the developer completes the preliminary ST for the composite product.
10. It is not mandatory that the composite product and its related base component are being evaluated according to the same edition of ISO/IEC 15408. 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 ISO/IEC 15408 series editions shall be established / acknowledged by the composite product evaluation authority.
11. If conformance to a PP is claimed, e.g. a composite product ST claims conformance to a PP (that possibly claims conformance to a further PP), the consistency check can be reduced to the elements of the ST 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”:
12. composite-ST → PP 1 → PP 2 ← base-ST
13. PP 1 may require any kind of conformance [[6]](#footnote-6), 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’.
14. Note that consistency may be no direct matching: Objectives for the base component’s environment may become objectives for the composite TOE.

### Consistency of Security Target (ST) (ASE\_COMP.1)

Developer action elements

ASE\_COMP.1.1D The developer shall provide a statement of compatibility between the composite product Security Target and the base component Security Target. This statement may be provided within the composite product Security Target.

Content and presentation elements

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.

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.

Evaluator action elements

ASE\_COMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Extended components definition (ASE\_ECD)

Objectives

1. Extended security requirements are requirements that are not based on components from ISO/IEC 15408-2 or this document, but which are based on extended components: components defined by the ST author.
2. Evaluation of the definition of extended components is necessary to determine that they are clear and unambiguous, and that they are necessary, i.e. they may not be clearly expressed using existing ISO/IEC 15408-2 or this document components.

### Extended components definition (ASE\_ECD.1)

Developer action elements

ASE\_ECD.1.1D The developer shall provide a statement of security requirements.

ASE\_ECD.1.2D The developer shall provide an extended components definition.

Content and presentation elements

ASE\_ECD.1.1C The statement of security requirements shall identify all extended security requirements.

ASE\_ECD.1.2C The extended components definition shall define an extended component for each extended security requirement.

ASE\_ECD.1.3C The extended components definition shall describe how each extended component is related to the existing ISO/IEC 15408 series components, families, and classes.

ASE\_ECD.1.4C The extended components definition shall use the existing ISO/IEC 15408 series components, families, classes, and methodology as a model for presentation.

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.

Evaluator action elements

ASE\_ECD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE\_ECD.1.2E The evaluator shall confirm that no extended component may be clearly expressed using existing components.

## ST introduction (ASE\_INT)

Objectives

1. The objective of this family is to describe the TOE in a narrative way on three levels of abstraction: TOE reference, TOE overview and TOE description.
2. Evaluation of the ST introduction is required to demonstrate that the ST and the TOE are correctly identified, that the TOE is correctly described at three levels of abstraction and that these three descriptions are consistent with each other.

### ST introduction (ASE\_INT.1)

Developer action elements

ASE\_INT.1.1D The developer shall provide an ST introduction.

Content and presentation elements

ASE\_INT.1.1C The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE\_INT.1.2C The ST reference shall uniquely identify the ST.

ASE\_INT.1.3C The TOE reference shall uniquely identify the TOE.

ASE\_INT.1.4C The TOE overview shall summarize the usage and major security features of the TOE.

ASE\_INT.1.5C The TOE overview shall identify the TOE type.

ASE\_INT.1.6C The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

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.

ASE\_INT.1.8C The TOE description shall describe the physical scope of the TOE.

ASE\_INT.1.9C The TOE description shall describe the logical scope of the TOE.

Evaluator action elements

ASE\_INT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE\_INT.1.2E The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.

## Security objectives (ASE\_OBJ)

Objectives

1. The security objectives are a concise statement of the intended response to the security problem defined through the [ASE\_SPD](#ase_spd) family.
2. Evaluation of the security objectives is required to demonstrate that the security objectives adequately and completely address the security problem definition, that the division of this problem between the TOE and its operational environment is clearly defined.

Levelling criteria

1. The components in this family are levelled on whether they prescribe only security objectives for the operational environment [ASE\_OBJ.1](#ase_obj.1), or also security objectives for the TOE [ASE\_OBJ.2](#ase_obj.2).

### Security objectives for the operational environment (ASE\_OBJ.1)

Developer action elements

ASE\_OBJ.1.1D The developer shall provide a statement of security objectives for the operational environment.

ASE\_OBJ.1.2D The developer shall provide a security objectives rationale for the operational environment.

Content and presentation elements

ASE\_OBJ.1.1C The statement of security objectives shall describe the security objectives for the operational environment.

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.

ASE\_OBJ.1.3C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

ASE\_OBJ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Security objectives (ASE\_OBJ.2)

Dependencies: [ASE\_SPD.1](#ase_spd.1)

Developer action elements

ASE\_OBJ.2.1D The developer shall provide a statement of security objectives.

ASE\_OBJ.2.2D The developer shall provide a security objectives rationale.

Content and presentation elements

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.

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.

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.

ASE\_OBJ.2.4C The security objectives **rationale** shall **demonstrate** **that** the security objectives **counter** **all** **threats.**

ASE\_OBJ.2.5C The security objectives rationale shall demonstrate that the security objectives enforce all OSPs.

ASE\_OBJ.2.6C The security objectives rationale shall demonstrate that the security objectives for the operational environment uphold all assumptions.

Evaluator action elements

ASE\_OBJ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Security requirements (ASE\_REQ)

Objectives

1. The SFRs form a clear, unambiguous and well-defined description of the expected security behaviour of the TOE. The SARs form a clear, unambiguous and canonical description of the expected activities that will be undertaken to gain assurance in the TOE.
2. Evaluation of the security requirements is required to ensure that they are clear, unambiguous and well-defined.

Levelling criteria

1. The components in this family are levelled on whether they are stated as is (see [ASE\_REQ.1](#ase_req.1)), or whether the SFRs are derived from security objectives for the TOE (see [ASE\_REQ.2](#ase_req.2).).

### Direct rationale security requirements (ASE\_REQ.1)

Dependencies: [ASE\_ECD.1](#ase_ecd.1)

Developer action elements

ASE\_REQ.1.1D The developer shall provide a statement of security requirements.

ASE\_REQ.1.2D The developer shall provide a security requirements rationale.

Content and presentation elements

ASE\_REQ.1.1C The statement of security requirements shall describe the SFRs and the SARs.

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.

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.

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.

ASE\_REQ.1.5C The statement of security requirements shall identify all operations on the security requirements.

ASE\_REQ.1.6C All operations shall be performed correctly.

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.

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.

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.

ASE\_REQ.1.10C The security requirements rationale shall explain why the SARs were chosen.

ASE\_REQ.1.11C The statement of security requirements shall be internally consistent.

ASE\_REQ.1.12C 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

Evaluator action elements

ASE\_REQ.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Derived security requirements (ASE\_REQ.2)

Dependencies: [ASE\_OBJ.2](#ase_obj.2), [ASE\_ECD.1](#ase_ecd.1)

Developer action elements

ASE\_REQ.2.1D The developer shall provide a statement of security requirements.

ASE\_REQ.2.2D The developer shall provide a security requirements rationale.

Content and presentation elements

ASE\_REQ.2.1C The statement of security requirements shall describe the SFRs and the SARs.

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.

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.

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.

ASE\_REQ.2.5C The statement of security requirements shall identify all operations on the security requirements.

ASE\_REQ.2.6C All operations shall be performed correctly.

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.

ASE\_REQ.2.8C The security requirements rationale shall demonstrate that the SFRs **meet** **all** security objectives for the TOE.

ASE\_REQ.2.9C The security requirements rationale shall explain why the SARs were chosen.

ASE\_REQ.2.10C The statement of security requirements shall be internally consistent.

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.

Evaluator action elements

ASE\_REQ.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Security problem definition (ASE\_SPD)

Objectives

1. This part of the ST defines the security problem to be addressed by the TOE and the operational environment of the TOE.
2. Evaluation of the security problem definition is required to demonstrate that the security problem intended to be addressed by the TOE and its operational environment, is clearly defined.

### Security problem definition (ASE\_SPD.1)

Developer action elements

ASE\_SPD.1.1D The developer shall provide a security problem definition.

Content and presentation elements

ASE\_SPD.1.1C The security problem definition shall describe the threats.

ASE\_SPD.1.2C All threats shall be described in terms of a threat agent, an asset, and an adverse action.

ASE\_SPD.1.3C The security problem definition shall describe the OSPs.

ASE\_SPD.1.4C The security problem definition shall describe the assumptions about the operational environment of the TOE.

Evaluator action elements

ASE\_SPD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## TOE summary specification (ASE\_TSS)

Objectives

1. The TOE summary specification enables evaluators and potential consumers to gain a general understanding of how the TOE is implemented.
2. Evaluation of the TOE summary specification is necessary to determine whether it is adequately described how the TOE:

* meets its SFRs;
* protects itself against interference, logical tampering and bypass;

1. and whether the TOE summary specification is consistent with other narrative descriptions of the TOE.

Levelling criteria

1. The components in this family are levelled on whether the TOE summary specification only needs to describe how the TOE meets the SFRs, or whether the TOE summary specification also needs to describe how the TOE protects itself against logical tampering and bypass. This additional description may be used in special circumstances where there can be a specific concern regarding the TOE security architecture.

### TOE summary specification (ASE\_TSS.1)

Dependencies: [ASE\_INT.1](#ase_int.1), [ASE\_REQ.1](#ase_req.1), [ADV\_FSP.1](#adv_fsp.1)

Developer action elements

ASE\_TSS.1.1D The developer shall provide a TOE summary specification.

Content and presentation elements

ASE\_TSS.1.1C The TOE summary specification shall describe how the TOE meets each SFR.

Evaluator action elements

ASE\_TSS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ASE\_TSS.1.2E The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

### TOE summary specification with architectural design summary (ASE\_TSS.2)

Dependencies: [ASE\_INT.1](#ase_int.1), [ASE\_REQ.1](#ase_req.1), [ADV\_ARC.1](#adv_arc.1)

Developer action elements

ASE\_TSS.2.1D The developer shall provide a TOE summary specification.

Content and presentation elements

ASE\_TSS.2.1C The TOE summary specification shall describe how the TOE meets each SFR.

ASE\_TSS.2.2C The TOE summary specification shall describe how the TOE protects itself against interference and logical tampering.

ASE\_TSS.2.3C The TOE summary specification shall describe how the TOE protects itself against bypass.

Evaluator action elements

ASE\_TSS.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

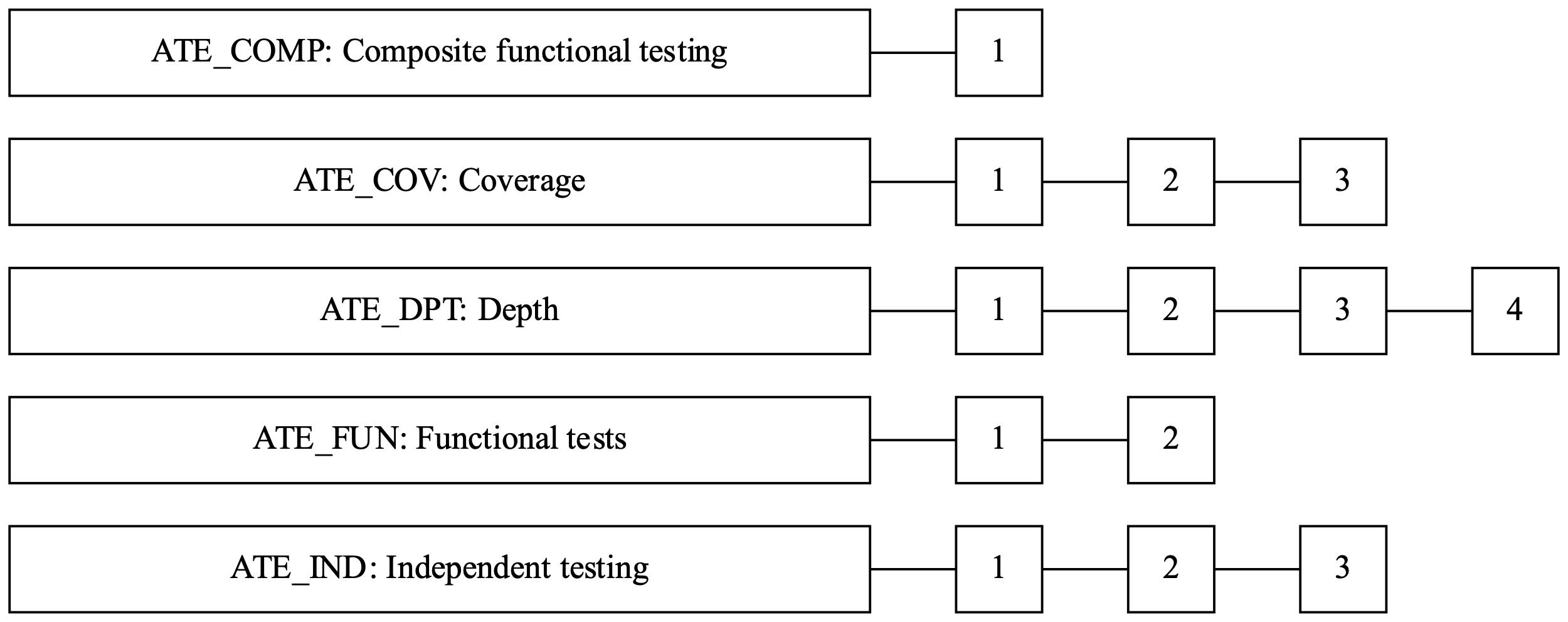
ASE\_TSS.2.2E The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

# Class ATE Tests

## Introduction

The class “Tests” encompasses five families: [ATE\_COV](#ate_cov), [ATE\_DPT](#ate_dpt), [ATE\_IND](#ate_ind) (i.e. functional testing performed by evaluators), [ATE\_FUN](#ate_fun) and [ATE\_COMP](#ate_comp). Testing provides assurance that the TSF behaves as described (in the functional specification, TOE design, implementation representation, and allows straightforward traceability of SFR in test scenario).

1. The emphasis in this class is on confirmation that the TSF operates according to its design descriptions. This class does not address penetration testing, which is based upon an analysis of the TSF that specifically seeks to identify vulnerabilities in the design and implementation of the TSF. Penetration testing is addressed separately as an aspect of vulnerability assessment in the [AVA](#ava) class.
2. The [ATE](#ate) class separates testing into developer testing and evaluator testing. The [ATE\_COV](#ate_cov), and [ATE\_DPT](#ate_dpt) families address the completeness of developer testing. [ATE\_COV](#ate_cov) addresses the rigour with which the functional specification is tested; [ATE\_DPT](#ate_dpt) addresses whether testing against other design descriptions (security architecture, TOE design, and implementation representation) is required.
3. [ATE\_FUN](#ate_fun) addresses the performing of the tests by the developer and how this testing should be documented. Finally, [ATE\_IND](#ate_ind) then addresses evaluator testing: whether the evaluator should repeat part or all of the developer testing and how much independent testing the evaluator should do.
4. [ATE\_COMP](#ate_comp) determines whether the composite product as a whole exhibits the properties necessary to satisfy the functional requirements of its ST.



|  | [ADV\_ARC.1](#adv_arc.1) | [ADV\_FSP.1](#adv_fsp.1) | [ADV\_FSP.2](#adv_fsp.2) | [ADV\_FSP.4](#adv_fsp.4) | [ADV\_IMP.1](#adv_imp.1) | [ADV\_TDS.2](#adv_tds.2) | [ADV\_TDS.3](#adv_tds.3) | [ADV\_TDS.4](#adv_tds.4) | [AGD\_OPE.1](#agd_ope.1) | [AGD\_PRE.1](#agd_pre.1) | [ATE\_COV.1](#ate_cov.1) | [ATE\_FUN.1](#ate_fun.1) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**ATE\_COMP.1**](#ate_comp.1) |  |  |  |  |  |  |  |  |  |  |  |  |
| [**ATE\_COV.1**](#ate_cov.1) |  |  | X |  |  |  |  |  |  |  | - | X |
| [**ATE\_COV.2**](#ate_cov.2) |  |  | X |  |  |  |  |  |  |  | - | X |
| [**ATE\_COV.3**](#ate_cov.3) |  |  | X |  |  |  |  |  |  |  | - | X |
| [**ATE\_DPT.1**](#ate_dpt.1) | X |  | - |  |  | X |  |  |  |  | - | X |
| [**ATE\_DPT.2**](#ate_dpt.2) | X |  | - |  |  |  | X |  |  |  | - | X |
| [**ATE\_DPT.3**](#ate_dpt.3) | X |  | - |  |  |  |  | X |  |  | - | X |
| [**ATE\_DPT.4**](#ate_dpt.4) | X |  | - |  | X |  |  | X |  |  | - | X |
| [**ATE\_FUN.1**](#ate_fun.1) |  |  | - |  |  |  |  |  |  |  | X | - |
| [**ATE\_FUN.2**](#ate_fun.2) |  |  | - |  |  |  |  |  |  |  | X | - |
| [**ATE\_IND.1**](#ate_ind.1) |  | X |  |  |  |  |  |  | X | X |  |  |
| [**ATE\_IND.2**](#ate_ind.2) |  |  | X |  |  |  |  |  | X | X | X | X |
| [**ATE\_IND.3**](#ate_ind.3) |  |  | - | X |  |  |  |  | X | X | X | X |

## Composite functional testing (ATE\_COMP)

Objectives

1. The aim of this family is to determine whether the composite product as a whole exhibits the properties necessary to satisfy the functional requirements of its composite product ST.

Levelling criteria

1. This family contains only one component.

Application notes

1. 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 its 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.
2. Integration testing means that the composite product is being tested as it is: the dependent component is running together with the related base component.
3. Some dependent component functionality testing can only be performed on emulators, before its embedding/integration onto 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](#ate) assurance class. No additional developer’s action is required here.
4. 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.
5. The behaviour of implementation of some SFRs can depend on properties of the base component as well as on 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.
6. This family 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](#ate). These integration tests shall be specified and performed, whereby the approach of the standard assurance families of the class [ATE](#ate) shall be applied.
7. The composite product evaluation sponsor shall ensure that the following is available for the composite product evaluator:

* composite product samples suitable for testing.

### Composite product functional testing (ATE\_COMP.1)

Developer action elements

ATE\_COMP.1.1D The developer shall provide a set of tests as required by the assurance package chosen.

ATE\_COMP.1.2D The developer shall provide the composite product for testing.

Content and presentation elements

ATE\_COMP.1.1C Content and presentation of the specification and documentation of the integration tests shall correspond to the standard requirements of the assurance families ATE\_FUN and ATE\_COV.

ATE\_COMP.1.2C The composite product provided shall be suitable for testing.

Evaluator action elements

ATE\_COMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Coverage (ATE\_COV)

Objectives

1. This family establishes that the TSF has been tested against its functional specification. This is achieved through an examination of developer evidence of correspondence.

Levelling criteria

1. The components in this family are levelled on the basis of specification.

### Evidence of coverage (ATE\_COV.1)

Dependencies: [ADV\_FSP.2](#adv_fsp.2), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_COV.1.1D The developer shall provide evidence of the test coverage.

Content and presentation elements

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.

Evaluator action elements

ATE\_COV.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Analysis of coverage (ATE\_COV.2)

Dependencies: [ADV\_FSP.2](#adv_fsp.2), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_COV.2.1D The developer shall provide **an** **analysis** of the test coverage.

Content and presentation elements

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.

ATE\_COV.2.2C The analysis of the test coverage shall demonstrate that all TSFIs in the functional specification have been tested.

Evaluator action elements

ATE\_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Rigorous analysis of coverage (ATE\_COV.3)

Dependencies: [ADV\_FSP.2](#adv_fsp.2), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_COV.3.1D The developer shall provide an analysis of the test coverage.

Content and presentation elements

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.

ATE\_COV.3.2C The analysis of the test coverage shall demonstrate that all TSFIs in the functional specification have been **completely** tested.

Evaluator action elements

ATE\_COV.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Depth (ATE\_DPT)

Objectives

1. The components in this family deal with the level of detail to which the TSF is tested by the developer. Testing of the TSF is based upon increasing depth of information derived from additional design representations and descriptions (TOE design, implementation representation, and security architecture description).
2. The objective is to counter the risk of missing an error in the development of the TOE. Testing that exercises specific internal interfaces can provide assurance not only that the TSF exhibits the desired external security behaviour, but also that this behaviour stems from correctly operating internal functionality.

Levelling criteria

1. The components in this family are levelled on the basis of increasing detail provided in the TSF representations, from the TOE design to the implementation representation. This levelling reflects the TSF representations presented in the [ADV](#adv) class.

Application notes

1. The TOE design describes the internal components (e.g. subsystems) and, perhaps, modules of the TSF, together with a description of the interfaces among these components and modules. Evidence of testing of this TOE design must show that the internal interfaces have been exercised and seen to behave as described. This may be achieved through testing via the external interfaces of the TSF, or by testing of the TOE subsystem or module interfaces in isolation, perhaps employing a test harness. In cases where some aspects of an internal interface cannot be tested via the external interfaces, there should either be justification that these aspects need not be tested, or the internal interface needs to be tested directly. In the latter case the TOE design needs to be sufficiently detailed in order to facilitate direct testing.
2. In cases where the description of the TSF’s architectural soundness (in [ADV\_ARC](#adv_arc)) cites specific mechanisms, the tests performed by the developer must show that the mechanisms have been exercised and seen to behave as described.
3. At the highest component of this family, the testing is performed not only against the TOE design, but also against the implementation representation.

### Testing: basic design (ATE\_DPT.1)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_TDS.2](#adv_tds.2), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_DPT.1.1D The developer shall provide the analysis of the depth of testing.

Content and presentation elements

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.

ATE\_DPT.1.2C The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.

Evaluator action elements

ATE\_DPT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Testing: security enforcing modules (ATE\_DPT.2)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_TDS.3](#adv_tds.3), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_DPT.2.1D The developer shall provide the analysis of the depth of testing.

Content and presentation elements

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.

ATE\_DPT.2.2C The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.

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.

Evaluator action elements

ATE\_DPT.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Testing: modular design (ATE\_DPT.3)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_TDS.4](#adv_tds.4), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_DPT.3.1D The developer shall provide the analysis of the depth of testing.

Content and presentation elements

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.

ATE\_DPT.3.2C The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.

ATE\_DPT.3.3C The analysis of the depth of testing shall demonstrate that **all** **TSF** modules in the TOE design have been tested.

Evaluator action elements

ATE\_DPT.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Testing: implementation representation (ATE\_DPT.4)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_TDS.4](#adv_tds.4), [ADV\_IMP.1](#adv_imp.1), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_DPT.4.1D The developer shall provide the analysis of the depth of testing.

Content and presentation elements

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.

ATE\_DPT.4.2C The analysis of the depth of testing shall demonstrate that all TSF subsystems in the TOE design have been tested.

ATE\_DPT.4.3C The analysis of the depth of testing shall demonstrate that all modules in the TOE design have been tested.

ATE\_DPT.4.4C The analysis of the depth of testing shall demonstrate that the TSF operates in accordance with its implementation representation.

Evaluator action elements

ATE\_DPT.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Functional tests (ATE\_FUN)

Objectives

1. Functional testing performed by the developer provides assurance that the tests in the test documentation are performed and documented correctly. The correspondence of these tests to the design descriptions of the TSF is achieved through the [ATE\_COV](#ate_cov) and [ATE\_DPT](#ate_dpt) families.
2. This family contributes to providing assurance that the likelihood of undiscovered flaws is relatively small.
3. The families [ATE\_COV](#ate_cov), [ATE\_DPT](#ate_dpt) and [ATE\_FUN](#ate_fun) are used in combination to define the evidence of testing to be supplied by a developer. Independent functional testing by the evaluator is specified by [ATE\_IND](#ate_ind).

Levelling criteria

1. This family contains two components, the higher requiring that ordering dependencies are analysed.

Application notes

1. Procedures for performing tests are expected to provide instructions for using test programs and test suites, including the test environment, test conditions, test data parameters and values. The test procedures should also show how the test results are derived from the test inputs.
2. Ordering dependencies are relevant when the successful execution of a particular test depends upon the existence of a particular state. For example, this can require that test A be executed immediately before test B, since the state resulting from the successful execution of test A is a prerequisite for the successful execution of test B. Thus, failure of test B can be related to a problem with the ordering dependencies. In the above example, test B can fail because test C (rather than test A) was executed immediately before it, or the failure of test B can be related to a failure of test A.

### Functional testing (ATE\_FUN.1)

Dependencies: [ATE\_COV.1](#ate_cov.1)

Developer action elements

ATE\_FUN.1.1D The developer shall test the TSF and document the results.

ATE\_FUN.1.2D The developer shall provide test documentation.

Content and presentation elements

ATE\_FUN.1.1C The test documentation shall consist of test plans, expected test results and actual test results.

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.

ATE\_FUN.1.3C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE\_FUN.1.4C The actual test results shall be consistent with the expected test results.

Evaluator action elements

ATE\_FUN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### Ordered functional testing (ATE\_FUN.2)

Dependencies: [ATE\_COV.1](#ate_cov.1)

Developer action elements

ATE\_FUN.2.1D The developer shall test the TSF and document the results.

ATE\_FUN.2.2D The developer shall provide test documentation.

Content and presentation elements

ATE\_FUN.2.1C The test documentation shall consist of test plans, expected test results and actual test results.

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.

ATE\_FUN.2.3C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE\_FUN.2.4C The actual test results shall be consistent with the expected test results.

ATE\_FUN.2.5C The test documentation shall include an analysis of the test procedure ordering dependencies.

Evaluator action elements

ATE\_FUN.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

## Independent testing (ATE\_IND)

Objectives

1. The objectives of this family are built upon the assurances achieved in the [ATE\_FUN](#ate_fun), [ATE\_COV](#ate_cov), and [ATE\_DPT](#ate_dpt) families by verifying the developer testing and performing additional tests by the evaluator.

Levelling criteria

1. Levelling is based upon the amount of developer test documentation and test support and the amount of evaluator testing.

Application notes

1. This family deals with the degree to which there is independent functional testing of the TSF. Independent functional testing may take the form of repeating the developer’s functional tests (in whole or in part) or of extending the scope or the depth of the developer’s tests. These activities are complementary, and an appropriate mix must be planned for each TOE, which takes into account the availability and coverage of test results, and the functional complexity of the TSF.
2. Sampling of developer tests is intended to provide confirmation that the developer has carried out his planned test programme on the TSF and has correctly recorded the results. The size of sample selected will be influenced by the detail and quality of the developer’s functional test results. The evaluator will also need to consider the scope for devising additional tests, and the relative benefit that may be gained from effort in these two areas. It is recognized that repetition of all developer tests may be feasible and desirable in some cases, but may be very arduous and less productive in others. The highest component in this family should therefore be used with caution. Sampling will address the whole range of test results available, including those supplied to meet the requirements of both [ATE\_COV](#ate_cov) and [ATE\_DPT](#ate_dpt).
3. There is also a need to consider the different configurations of the TOE that are included within the evaluation. The evaluator will need to assess the applicability of the results provided, and to plan his own testing accordingly.
4. The suitability of the TOE for testing is based on the access to the TOE, and the supporting documentation and information required (including any test software or tools) to run tests. The need for such support is addressed by the dependencies to other assurance families.
5. Additionally, suitability of the TOE for testing may be based on other considerations. For example, the version of the TOE submitted by the developer may not be the final version.
6. The term interfaces refers to interfaces described in the functional specification and TOE design, and parameters passed through invocations identified in the implementation representation. The exact set of interfaces to be used is selected through [ATE\_COV](#ate_cov) and the [ATE\_DPT](#ate_dpt) components.
7. References to a subset of the interfaces are intended to allow the evaluator to design an appropriate set of tests which is consistent with the objectives of the evaluation being conducted.

### Independent testing - conformance (ATE\_IND.1)

Dependencies: [ADV\_FSP.1](#adv_fsp.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1)

Developer action elements

ATE\_IND.1.1D The developer shall provide the TOE for testing.

Content and presentation elements

ATE\_IND.1.1C The TOE shall be suitable for testing.

Evaluator action elements

ATE\_IND.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE\_IND.1.2E The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

### Independent testing - sample (ATE\_IND.2)

Dependencies: [ADV\_FSP.2](#adv_fsp.2), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1), [ATE\_COV.1](#ate_cov.1), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_IND.2.1D The developer shall provide the TOE for testing.

Content and presentation elements

ATE\_IND.2.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

ATE\_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE\_IND.2.2E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

ATE\_IND.2.3E The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

### Independent testing - complete (ATE\_IND.3)

Dependencies: [ADV\_FSP.4](#adv_fsp.4), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1), [ATE\_COV.1](#ate_cov.1), [ATE\_FUN.1](#ate_fun.1)

Developer action elements

ATE\_IND.3.1D The developer shall provide the TOE for testing.

Content and presentation elements

ATE\_IND.3.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

ATE\_IND.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

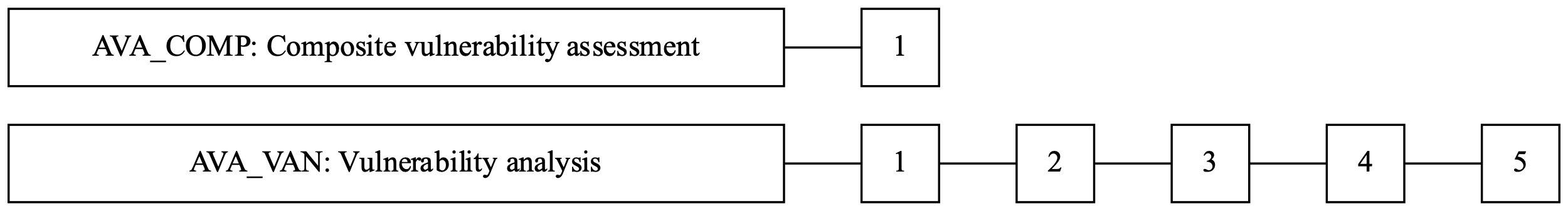
ATE\_IND.3.2E The evaluator shall execute **all** tests in the test documentation to verify the developer test results.

ATE\_IND.3.3E The evaluator shall test the TSF to confirm that the **entire** TSF operates as specified.

# Class AVA Vulnerability assessment

## Introduction

The [AVA](#ava) class addresses the possibility of exploitable vulnerabilities introduced in the development or the operation of the TOE.



|  | [ADV\_ARC.1](#adv_arc.1) | [ADV\_FSP.1](#adv_fsp.1) | [ADV\_FSP.2](#adv_fsp.2) | [ADV\_FSP.4](#adv_fsp.4) | [ADV\_IMP.1](#adv_imp.1) | [ADV\_TDS.1](#adv_tds.1) | [ADV\_TDS.3](#adv_tds.3) | [AGD\_OPE.1](#agd_ope.1) | [AGD\_PRE.1](#agd_pre.1) | [ATE\_DPT.1](#ate_dpt.1) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**AVA\_COMP.1**](#ava_comp.1) |  |  |  |  |  |  |  |  |  |  |
| [**AVA\_VAN.1**](#ava_van.1) |  | X |  |  |  |  |  | X | X |  |
| [**AVA\_VAN.2**](#ava_van.2) | X |  | X |  |  | X |  | X | X |  |
| [**AVA\_VAN.3**](#ava_van.3) | X |  |  | X | X |  | X | X | X | X |
| [**AVA\_VAN.4**](#ava_van.4) | X |  |  | X | X |  | X | X | X | X |
| [**AVA\_VAN.5**](#ava_van.5) | X |  |  | X | X |  | X | X | X | X |

## Application notes

Generally, the vulnerability assessment activity covers various vulnerabilities in the development and operation of the TOE. Development vulnerabilities take advantage of some property of the TOE, or the product where the TOE resides, which was introduced during its development, e.g. defeating the TSF self-protection through tampering, direct attack or monitoring of the TSF, defeating the TSF domain separation through monitoring or direct attack the TSF, or defeating non-bypassability through circumventing (bypassing) the TSF. Explicit dependencies of the TOE on IT systems in the environment must also be considered. Operational vulnerabilities take advantage of weaknesses in non-technical countermeasures to violate the TOE SFRs, e.g. misuse or incorrect configuration. Misuse investigates whether the TOE can be configured or used in a manner that is insecure, but that an administrator or user of the TOE would reasonably believe to be secure.

1. Assessment of development vulnerabilities is covered by the assurance family [AVA\_VAN](#ava_van). Basically, all development vulnerabilities can be considered in the context of [AVA\_VAN](#ava_van) due to the fact, that this family allows application of a wide range of assessment methodologies being unspecific to the kind of an attack scenario. These unspecific assessment methodologies comprise, among other, also the specific methodologies for those TSF where covert channels are to be considered (a channel capacity estimation can be done using informal engineering measurements, as well as actual test measurements) or can be overcome by the use of sufficient resources in the form of a direct attack (underlying technical concept of those TSF is based on probabilistic or permutational mechanisms; a qualification of their security behaviour and the effort required to overcome them can be made using a quantitative or statistical analysis).
2. If there are security objectives specified in the ST to either to prevent one user of the TOE from observing activity associated with another user of the TOE, or to ensure that information flows cannot be used to achieve enforced illicit data signals, covert channel analysis should be considered during the conduct of the vulnerability analysis. This is often reflected by the inclusion of Unobservability (FPR\_UNO) and multilevel access control policies specified through Access control policy (FDP\_ACC) and/or Information flow control policy (FDP\_IFC) requirements in the ST.

## Composite vulnerability assessment (AVA\_COMP)

Objectives

1. The aim of this family is to determine the exploitability of flaws or weaknesses in the composite product as a whole in the intended environment.

Levelling criteria

1. This family contains only one component.

Application notes

1. This family 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 [[7]](#footnote-7) assurance family of the class [AVA](#ava): [AVA\_VAN](#ava_van).
2. The composite product evaluator shall perform a vulnerability analysis for the composite product using, amongst other, the results of the base component evaluation. This vulnerability analysis shall be confirmed by penetration testing.
3. 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](#alc_dvs).
4. 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 stipulations provided with the base component. Different possibilities exist to shorten composite product vulnerability analysis in such cases: E.g. 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 on 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 on his own discretion. The intention hereby is to use test dependent components without countermeasures and without deactivating any base component inherent countermeasure.
5. The results of the vulnerability assessment for the base component of the composite product represented in the ETR for composite evaluation can be re-used under the following conditions: they are up-to-date and all composite activities for correctness – [ASE\_COMP.1](#ase_comp.1), [ALC\_COMP.1](#alc_comp.1), [ADV\_COMP.1](#adv_comp.1) and [ATE\_COMP.1](#ate_comp.1) – are finalised with the verdict PASS.
6. Due to composing of the base component and the dependent component a new quality arises, which may cause additional vulnerabilities of the base component which might be not mentioned 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.
7. 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 report of the base component evaluation authority.

### Composite product vulnerability assessment (AVA\_COMP.1)

Developer action elements

AVA\_COMP.1.1D The developer shall provide the composite product for penetration testing.

Content and presentation elements

AVA\_COMP.1.1C The composite product provided shall be suitable for testing as a whole.

Evaluator action elements

AVA\_COMP.1.1E The evaluator shall conduct penetration testing of the composite product as a whole building on the evaluator’s own vulnerability analysis to ensure that the vulnerabilities being relevant for the composite product Security Target are not exploitable.

## Vulnerability analysis (AVA\_VAN)

Objectives

1. Vulnerability analysis is an assessment to determine whether potential vulnerabilities identified, during the evaluation of the development and anticipated operation of the TOE or by other methods (e.g. by flaw hypotheses or quantitative or statistical analysis of the security behaviour of the underlying security mechanisms), can allow attackers to violate the SFRs.
2. Vulnerability analysis deals with the threats that an attacker will be able to discover flaws that will allow unauthorised access to data and functionality, allow the ability to interfere with or alter the TSF, or interfere with the authorized capabilities of other users.
3. In case of a multi-assurance evaluation the vulnerability analysis shall assess the defined sub-TSF as well as the TOE as a whole.

Levelling criteria

1. Levelling is based on an increasing rigour of vulnerability analysis by the evaluator and increased levels of attack potential required by an attacker to identify and exploit the potential vulnerabilities.

### Vulnerability survey (AVA\_VAN.1)

Dependencies: [ADV\_FSP.1](#adv_fsp.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1)

Developer action elements

AVA\_VAN.1.1D The developer shall provide the TOE for testing.

Content and presentation elements

AVA\_VAN.1.1C The TOE shall be suitable for testing.

Evaluator action elements

AVA\_VAN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA\_VAN.1.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA\_VAN.1.3E The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

### Vulnerability analysis (AVA\_VAN.2)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_FSP.2](#adv_fsp.2), [ADV\_TDS.1](#adv_tds.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1)

Developer action elements

AVA\_VAN.2.1D The developer shall provide the TOE for testing.

AVA\_VAN.2.2D The developer shall provide a list of third party components included in the TOE and the TOE delivery.

Content and presentation elements

AVA\_VAN.2.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

AVA\_VAN.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA\_VAN.2.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE the components in the list of third party components, and specific IT products in the environment that the TOE depends on.

AVA\_VAN.2.3E The evaluator shall perform **an** **independent** **vulnerability** **analysis** of **the** **TOE** **using** **the** **guidance** **documentation,** **functional** **specification,** **TOE** **design** **and** **security** **architecture** **description** to identify potential vulnerabilities in the TOE.

AVA\_VAN.2.4E The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

### Focused vulnerability analysis (AVA\_VAN.3)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_FSP.4](#adv_fsp.4), [ADV\_TDS.3](#adv_tds.3), [ADV\_IMP.1](#adv_imp.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1), [ATE\_DPT.1](#ate_dpt.1)

Developer action elements

AVA\_VAN.3.1D The developer shall provide the TOE for testing.

AVA\_VAN.3.2D The developer shall provide a list of third party components included in the TOE and the TOE delivery.

Content and presentation elements

AVA\_VAN.3.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

AVA\_VAN.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA\_VAN.3.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE the components in the list of third party components, and specific IT products in the environment that the TOE depends on.

AVA\_VAN.3.3E The evaluator shall perform an independent, **focused** vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description **and** **implementation** **representation** to identify potential vulnerabilities in the TOE.

AVA\_VAN.3.4E The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing **Enhanced-Basic** attack potential.

### Methodical vulnerability analysis (AVA\_VAN.4)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_FSP.4](#adv_fsp.4), [ADV\_TDS.3](#adv_tds.3), [ADV\_IMP.1](#adv_imp.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1), [ATE\_DPT.1](#ate_dpt.1)

Developer action elements

AVA\_VAN.4.1D The developer shall provide the TOE for testing.

AVA\_VAN.4.2D The developer shall provide a list of third party components included in the TOE and the TOE delivery.

Content and presentation elements

AVA\_VAN.4.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

AVA\_VAN.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA\_VAN.4.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE the components in the list of third party components, and specific IT products in the environment that the TOE depends on.

AVA\_VAN.4.3E The evaluator shall perform an independent, **methodical** vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify potential vulnerabilities in the TOE.

AVA\_VAN.4.4E The evaluator shall conduct penetration testing based on the identified potential vulnerabilities to determine that the TOE is resistant to attacks performed by an attacker possessing **Moderate** attack potential.

### Advanced methodical vulnerability analysis (AVA\_VAN.5)

Dependencies: [ADV\_ARC.1](#adv_arc.1), [ADV\_FSP.4](#adv_fsp.4), [ADV\_TDS.3](#adv_tds.3), [ADV\_IMP.1](#adv_imp.1), [AGD\_OPE.1](#agd_ope.1), [AGD\_PRE.1](#agd_pre.1), [ATE\_DPT.1](#ate_dpt.1)

Developer action elements

AVA\_VAN.5.1D The developer shall provide the TOE for testing.

AVA\_VAN.5.2D The developer shall provide a list of third party components included in the TOE and the TOE delivery.

Content and presentation elements

AVA\_VAN.5.1C The TOE shall be suitable for testing.

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.

Evaluator action elements

AVA\_VAN.5.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA\_VAN.5.2E The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE the components in the list of third party components, and specific IT products in the environment that the TOE depends on.

AVA\_VAN.5.3E The evaluator shall perform an independent, methodical vulnerability analysis of the TOE using the guidance documentation, functional specification, TOE design, security architecture description and implementation representation to identify potential vulnerabilities in the TOE.

AVA\_VAN.5.4E The evaluator shall conduct penetration testing based on the identified potential vulnerabilities to determine that the TOE is resistant to attacks performed by an attacker possessing **High** attack potential.

1. Denoted by composite product Security Target or composite-ST in the following. [↑](#footnote-ref-1)
2. Denoted by base component Security Target or base-ST in the following. [↑](#footnote-ref-2)
3. Generally, a Security Target expresses a security policy for the TOE defined. [↑](#footnote-ref-3)
4. Because the TSF enforce the Security Target (together with the organisational measures enforcing the security objectives for the operational environment of the TOE). [↑](#footnote-ref-4)
5. 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. [↑](#footnote-ref-5)
6. e.g. “strict”, “exact” or “demonstrable” according to the ISO/IEC 15408 series. [↑](#footnote-ref-6)
7. i.e. as defined by ISO/IEC 18045. [↑](#footnote-ref-7)