



Common Criteria Evaluation and Validation Scheme

Transition Guide

Summary of changes in the CC/CEM from v2.3 to v3.1

4 September 2008

NIAP/CCEVS
9800 Savage Road, STE 6757,
Ft. Meade, MD 20755-6757
Phone: (410) 854-4458
E-mail: scheme-comments@missi.ncsc.mil
<http://www.niap-ccevs.org/cc-scheme>

Table of Contents

1.	Overview of Common Criteria Changes.....	3
1.1.	Part 1- Introduction and general model.....	3
1.2.	Part 2- Security functional components.....	3
1.3.	Part 3- Security assurance components	3
a.	ASE/APE - Security Target evaluation / Protection Profile evaluation.....	3
b.	ACM/ADO/AGD/ALC – Life Cycle Support / Guidance Documentation.....	4
c.	ADV – Development	4
d.	ATE – Test.....	5
e.	AVA - Vulnerability Assessment	5
1.4.	CEM - Evaluation methodology	5
2	Mapping of Changes to Assurance Components	6
2.1.	Mapping of ASE/APE requirements.....	6
2.2.	Mapping of ACM/ADO/ALC/AGD requirements.....	7
2.3.	Mapping of ADV requirements	8
2.4.	Mapping of ATE requirements.....	10
2.5.	Mapping of AVA requirements	11

1. Overview of Common Criteria Changes

1.1. *Part 1- Introduction and general model*

Part 1 was made clearer, better reflecting the whole CC. This primarily consists of establishing and using a more consistent terminology, and to reflect the changes that were made to ASE/APE.

1.2. *Part 2- Security functional components*

Part 2 has had little changes from v2.3 to v3.1. The only notable change is the deletion of FPT_SEP and FPT_RVM. These were deleted as optional SFRs on the grounds that they describe attributes of any but the lowest assurance TSFs, and have been replaced by the ADV_ARC assurance requirement family. The descriptions of how the TSF cannot be bypassed and how it protects itself (FPT_RVM and FPT_SEP, respectively) *should* be sufficient for the description of how the TSF is of a sound architecture (ADV_ARC). Past evaluations have suffered from the lack of a comprehensive approach to addressing this critical principle, so the ADV_ARC requirements were crafted to mandate specific evidence be provided for analysis.

1.3. *Part 3- Security assurance components*

The changes to part 3, and their causes, are so varied that it would not be useful to attempt a consolidated global summary. Instead, the classes, or groups thereof, are discussed.

a. **ASE/APE - Security Target evaluation / Protection Profile evaluation**

ASE and APE contained numerous instances where elements were stated such that work was repeated. In addition, there was insufficient guidance on determining the adequacy of Assumption, Threat, OSP, or Security Objectives statements; consequently, this text could be determined to pass, yet prove to be useless for end users (i.e. potential customers).

The approach taken in rewriting these families was to organize the descriptions to yield a useful resulting ST/PP, while streamline the work in evaluating it. Included in this reorganization was providing descriptions of good Assumption, Threat, OSP, and Security Objectives statements, as well as focusing on the fact that the purpose of the TOE Summary Specification is to explain how the TOE meets its requirements.

A summary mapping of the PP contents as specified in versions 2.3 and 3.1 can be found in Annex 1. In this table, the contents as

defined in Figure 11 (v2.3) and Figure 9 (v3.1) of Part 1 have been put into outline format, and then mapped.

b. ACM/ADO/AGD/ALC – Life Cycle Support / Guidance Documentation

ACM/ADO/AGD/ALC were, for the most part, just rearranging the contents. The primary problem with v2.3 was that there was no clear delineation among them. For example, the configuration management requirements addressed in ACM should be in place over the entire lifecycle of the TOE, which is in fact the subject of ALC; and the actions required by the administrator (which are described in AGD) might also include actions associated with the start-up of the TOE, which is part of ADO.

These four classes were therefore rearranged into two: ALC which addresses the requirements associated with the developer's site, and AGD which addresses all of the requirements associated with the user's/ customer's site.

A summary mapping of the ACM/ADO/AGD/ALC requirements of versions 2.3 and 3.1 can be found in Annex 3.

c. ADV – Development

The problems with v2.3 of ADV were varied. In some cases it called for added work that was inefficient (RCR calling for a separate correspondence that would be much more straightforward if incorporated into the other families, à la v1.0). In some cases, it reflected a technology bias that the CC purported not to have (the two levels of abstraction approach in HLD/LLD is infeasible for very complex TOEs and unnecessary for very simple TOEs). In some cases, the components were not granular enough to allow the assurance to track along with the EAL scale (FSP remained unchanged from EAL1 through EAL3). Some basic IT security principles were completely missing (the absence of an architecture argument meant that all of the claimed security functions could be corrupted or bypassed, making them meaningless). And in all cases, there was no acknowledgement that some parts of the TSF are more critical and security-interesting than others, the result of which was that the analysis had to be performed -- even on parts that no security professional would bother with -- thereby expending vast amount of unnecessary effort.

The 3.1 version of ADV reflects a reasonable scale of increasing assurance with a corresponding amount of work. Although version 3.1 has more text (elements in the components and, hence, more work units in the methodology), this does not mean more work for evaluators/developers/certifiers; in fact, a considerable amount of the text describes both the principles underlying security analysis as well as how to *save* effort in performing it.

A summary mapping of the ADV requirements of versions 2.3 and 3.1 can be found in Annex 4.

d. ATE – Test

ATE was updated to reflect the fact that COV is tied to the testing of TSFI (and is therefore tied to the functional specification), while DPT was updated to reflect the levels of TDS.

A summary mapping of the ATE requirements of versions 2.3 and 3.1 can be found in Annex 5.

e. AVA - Vulnerability Assessment

The AVA class consists of only one family in v3.1 – AVA_VAN. AVA has removed the explicit notion of SOF (to reflect there is no longer a separate SOF claim made in PPs/STs). However, the evaluator should consider the resistance to attack of security functions in AVA_VAN. Most of the MSU analysis has been merged into the AGD family (because it simply extends the requirements of the quality of those documents). Finally, it created a new lowest level of vulnerability analysis, based upon the public domain and removed the requirements that the developer performs a vulnerability assessment; note that vulnerability analysis now bears the trigraph "VAN".

A summary mapping of the AVA requirements of versions 2.3 and 3.1 can be found in Annex 2.

1.4. *CEM - Evaluation methodology*

The CEM was rearranged into class/family/component, echoing the structure of Part 3, rather than being arranged by EAL as the previous version had been. It also was not limited to the components that make up EALs 1-4: if methodology for any higher-level component was available, it is included. However, mutual recognition is still limited to EAL4.

2 Mapping of Changes to Assurance Components

2.1. Mapping of ASE/APE requirements

v3.1	v2.3
1. PP Introduction A. PP reference	1. PP Introduction A. PP identification B. PP overview
B. TOE overview	2. TOE Description
2. Conformance Claims A. CC Conformance Claim B. PP Claim, Package Claim C. Conformance Rationale D. Conformance Statement	[As defined in section 5.4, as modified (and re-titled “Conformance Results”) by interpretation CCIMB-0008]
3. Security Problem Definition A. Threats B. OSPs C. Assumptions	3. TOE Security Environment A. Assumptions B. Threats C. OSPs
4. Security Objectives A. SOs for the TOE B. SOs for the operational environment	4. Security Objectives A. SOs for the TOE B. SOs for the environment
C. Security Objectives rationale	(8A, below)
5. Extended Components Definition	(5A1 and 5A2, below: the explicit reqs)
6. Security Requirements A. SFRs for the TOE	5. IT Security Requirements A. TOE Security Requirements 1. TOE SFRs [Part 2 <i>and</i> explicit reqs]
B. SARs for the TOE	2. TOE SARs [Part 3 <i>and</i> explicit reqs]
C. Security requirements rationale (requirements for environment are now optional)	(8B, below) B. Security Requirements for IT environment
(no separate App notes section; these can be put into Intro)	7. PP Application Notes
[see 4C, above]	8. Rationale A. Security Objectives Rationale
[see 6C, above]	B. Security Requirements Rationale

2.2. Mapping of ACM/ADO/ALC/AGD requirements

CC v2.3	CC v3.1
ACM_SCP – what is tracked by CM	ALC_CMS – scope of CM: what is covered ALC_CMC – capabilities of CM system (including whether automated)
ACM_CAP – a CM system; its capabilities; maintenance of CIs	
ACM_AUT – automated CM	
ALC_DVS – developer security	ALC_DVS – developer security
ALC_FLR – flaw remediation	ALC_FLR – flaw remediation
ALC_LCD – lifecycle development	ALC_LCD – lifecycle development
ALC_TAT – tools and techniques	ALC_TAT – tools and techniques
ADO_DEL – delivery procedures (at both developer’s site and user’s site)	ALC_DEL – delivery procedures (at the developer’s site) [user-side moved to AGD_PRE]
ADO_IGS – installation, generation, start-up procedures (at both developer’s site and user’s site)	AGD_PRE – preparation of TOE at the user’s site: User-side delivery procedures (receipt); User-side start-up procedures; [developer-side generation procedures moved to ALC_CMC]; subject to misuse analysis (formerly AVA_MSU)
AGD_ADM	AGD_OPE – operation: guidance on how to operate the TOE, aimed at humans that interact with it; subject to misuse analysis (formerly AVA_MSU)
AGD_USR	

2.3. Mapping of ADV requirements

CC v2.3	CC v3.1
<u>Families Addressing Decomposition of TSF</u>	
[no v.21 equivalent]	FSP.1 allege security-enforcing interfaces
[no v.21 equivalent]	FSP.2 describe security-enforcing interfaces
[no v.21 equivalent]	FSP.3 describe security-relevant interfaces
FSP.1 describe all interfaces	[deleted: covered by FSP.1, FSP.2, and FSP.3 to address varying levels of assurance at EAL1-3]
FSP.2 give all details of all interfaces	FSP.4 give all details of all interfaces ¹
FSP.3 give semiformal presentation	FSP.5 give semiformal presentation
FSP.4 give formal presentation	FSP.6 give formal presentation
HLD – high level description of TSF, regardless of assurance level	TDS – a high-level description of TSF at low assurance levels, migrating toward a more detailed description as assurance increases
LLD – low-level description of TSF, regardless of assurance level	
IMP.1 implementation subset provided and examined	IMP.1 entire implementation available; subset examined; sample mapped to design description
IMP.2 entire implementation provided	IMP.2 entire implementation available; subset examined; completely mapped to design description
IMP.3 structured implementation	[deleted: covered by INT]
RCR.1 informal correspondence RCR.2 semiformal correspondence RCR.3 formal correspondence	[correspondence is distributed through families: each representation must demonstrate correspondence to the previous one.]
<u>Families Addressing Understandability and Soundness of TSF</u>	
FPT_SEP, FPT_RVM SFRs	ARC - describe architectural soundness and mechanisms used to ensure domain isolation and nonbypassability
[no v.21 equivalent]	INT.1 well structured specified subset of TSF
INT.1 modularity of TSF	INT.2 well structured internals
INT.2 reduced of complexity: layers	[deleted: notion of layers is implicit in structure and complexity]
INT.3 minimal complexity	INT.3 minimally complex internals

¹ While this table gives the impression that FSP.2 is equivalent to FSP.4, this is not the case. There are additional aspects of the interface that introduced in v3.1 that do not exist in v2.3 (e.g., different types of error, and method of use).

SPM.1 informal model	[deleted; the informal SPM is the collection of Objectives in the ST]
SPM.2 semiformal model	[deleted; the semiformal SPM is the collection of SFRs in the ST]
SPM.3 formal model	SPM.1 formal model

2.4. Mapping of ATE requirements

CC v2.3	CC v3.1
ATE_COV.1 – test TSFIs	ATE_COV.1 – test TSFIs in the functional specification
ATE_COV.2 – test all TSFIs	ATE_COV.1 – analysis demonstrates all TSFI are tested in accordance with the functional specification
ATE_COV.3 – completely test all TSFIs	ATE_COV.1 – rigorous analysis demonstrates all TSFI are completely tested
ATE_DPT.1 – test against high-level design	ATE_DPT.1 – test TSF subsystems
	ATE_DPT.2 – test TSF subsystems and SFR-enforcing modules
ATE_DPT.2 – test against low-level design	ATE_DPT.3 – test TSF subsystems and modules
ATE_DPT.3 – test against implementation	ATE_DPT.4 – test against implementation
ATE_FUN.1 – functional testing	ATE_FUN.1 – functional testing
ATE_FUN.2 – ordered functional testing	ATE_FUN.2 – ordered functional testing
ATE_IND.1 – evaluator tests subset of TSF	ATE_IND.1 – evaluator tests subset of TSF
ATE_IND.2 – evaluator runs sample of developer’s tests	ATE_IND.2 – evaluator runs sample of developer’s tests
ATE_IND.3 – evaluator runs all of developer’s tests	ATE_IND.3 – evaluator runs all of developer’s tests

2.5. Mapping of AVA requirements

CC v2.3	CC v3.1
AVA_CCA – covert channel analysis	Covert channel analysis (which is relevant only when multi-level information flow or unobservability policies are present) moved into VAN.
AVA_MSU – misuse analysis: how might the documentation be misinterpreted in a way that leads to insecure use?	Misuse analysis mostly moved into the AGD families that address the documents subject to such analysis. However, aspects of this remain in VAN, since the AGD documents are considered when performing vulnerability analysis.
AVA_SOF – strength of function analysis: how strong are the permutational/probabilistic mechanisms?	SOF analysis no longer explicitly addressed. The resistance of a security function to an attacker is covered in VAN; no more SOF claim made
AVA_VLA – vulnerability analysis	AVA_VAN – vulnerability analysis: explicitly added the requirement look to public domain sources of vulnerabilities. Removed the notion of the developer performing any vulnerability assessment.