D21.3

Trustworthy MILS: CC Composite Evaluation Approach

Project number:	318353
Project acronym:	EURO-MILS
Project title:	EURO-MILS: Secure European Virtualisation for Trustworthy Applications in Critical Domains
Start date of the project:	1st October, 2012
Duration:	42 months
Programme:	FP7/2007-2013

Deliverable type:	Report
Deliverable reference number:	ICT-318353 / D21.3
Activity and Work package contributing to the deliverable:	Activity 2 / WP 21
Due date:	31st March, 2015
Actual submission date:	13th April, 2015

Responsible organisation:	EADS IW / Airbus Group
Responsible contract person:	Kevin Müller
Dissemination level:	Public
Revision:	1.0

Abstract:	As high assurance software systems are becoming more complex and sophisticated, assuring their security and safety is increasingly difficult and costly. Mono-lithic evaluation approaches do not scale well because evaluation effort grows exponentially with the complexity of the evaluation target. To keep pace with growing assurance demands, a compositional evaluation approach is a promising strategy. In a compositional evaluation, the individual components of a system are evaluated independently, and these partial evaluation results are composed to derive the overall evaluation verdict with minimum additional effort. The Common Criteria for IT Security Evaluation

	(ISO/IEC 15408) and the sup-porting documentation offer two different compositional evaluation schemes: the “Composite Product Evaluation for Smart Cards and Similar Devices” (CPE) and the “Composed Assurance Package” (CAP). In this report, we assess the suitability of CPE in the avionics domain, and we compare this evaluation scheme with its CAP alternative. We use the problem of evaluating an avionic security gateway as a case study to illustrate the implications, advantages, and drawbacks of the CPE approach.
Keywords:	Study, IT Security, Common Criteria, Compositional Security Evaluation, Composite Product Evaluation

Editor

Reinhard Schwarz (Fraunhofer IESE on behalf of EADS IW)

Kevin Müller (EADS IW)

Axel Söding-Freiherr von Blomberg (OPSYN)

Contributors (ordered according to beneficiary numbers)

Bertrand Leconte (AOS), Gobbo Gilles (AOS)

Reinhard Schwarz (Fraunhofer IESE)

Kevin Müller, Michael Paulitsch (EADS IW)

Axel Söding-Freiherr von Blomberg (OPSYN)

Axel Tillequin (EADS F IW)

Disclaimer

“This project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement n° 318353.”

This document has gone through the consortium’s internal review process and is still subject to the review of the European Commission. Updates to the content may be made at a later stage.

Executive Summary

As high assurance software systems are becoming more complex and sophisticated, assuring their security and safety is increasingly difficult and costly. Mono-lithic evaluation approaches do not scale well because evaluation effort grows exponentially with the complexity of the evaluation target. To keep pace with growing assurance demands, a compositional evaluation approach is a promising strategy.

In a compositional evaluation, the individual components of a system are evaluated independently, and these partial evaluation results are composed to derive the overall evaluation verdict with minimum additional effort. The Common Criteria for IT Security Evaluation (ISO/IEC 15408) and the sup-porting documentation offer two different compositional evaluation schemes: the “Composite Product Evaluation for Smart Cards and Similar Devices” (CPE) and the “Composed Assurance Package” (CAP).

In this report, we assess the suitability of CPE in the avionics domain, and we compare this evaluation scheme with its CAP alternative. We use the problem of evaluating an avionic security gateway as a case study to illustrate the implications, advantages, and drawbacks of the CPE approach.

Table of Content

List of Figures

List of Tables

Chapter 1

The Need for compositional Security Evaluation

A modern aircraft as well as an automobile is a complex, interconnected system, equipped with a plethora of hardware and software components. For aircrafts such systems control essential functions such as flight attitude, navigation, and communication, but also less critical convenience functions such as on-board entertainment, heating, ventilation, and air conditioning. To satisfy the growing demand for safe, reliable, convenient, environment-friendly, and economic air transportation, avionics are becoming ever more complex and sophisticated. Correspondingly, assuring their safety and security is increasingly difficult, time-consuming, and costly.

1.1 The Role of Security Evaluations for Flight Safety

Among the qualities of modern, software-controlled aircraft components, safety is paramount. However, as hardware and software integration of avionics advances and on-board functionalities increasingly depend on secure communication with off-board systems, IT security is gaining importance for flight safety. Due to the strong cross-linking and interdependence among flight control components, any unintentionally or deliberately misbehaving on-board or off-board IT component might—in the worst case—spoil the safety design of the whole aircraft. Therefore, IT security assurance is becoming an indispensable part of an aircraft's safety approval.

Unfortunately, IT security assurance is an intricate problem. Accordingly, security assurance will likely become a major obstacle for cost-efficient and timely safety approval of contemporary avionics, unless the efficiency of security evaluations can be improved significantly in the future.

1.2 The Role of Security Evaluations for Automotive

A strong tendency in the automotive market is the provision of online services within vehicles. While the market requests these features the impact to (in)security has been under-estimated. Various examples of cars getting hacked through connectivity interfaces to take remote control of critical functions are available on the internet.

While functional safety has been handled well by the automotive industry for some decades (e.g. ISO-26262), security has not. By introducing new attack vectors with technologies like eCall modem or G3/G4 connectivity, an increasing effort for security considerations is mandatory to achieve safety. However, since the automotive market is extremely price-driven, classic security evaluation methods miscarry simply on the high effort needed to evaluate these complex systems.

Another relevant aspect is the lifetime of vehicles, which can easily extend 20 years. In order to make software secure over such cycles, software updates is an important security feature. However, due to cost reasons it is hardly possible to re-evaluate the entire system after fixing issues only affecting a subset of the system's functionality. A compositional evaluation approach seems to be the only practical alternative.

1.3 Monolithic Security Evaluation

Today, the standard approach for security evaluations of IT products is a monolithic assessment, where the target of evaluation is studied in its entirety and in one sweep. The *Common Criteria for IT Security Evaluation* (CC) [CC12] is an established, internationally recognized evaluation standard. A companion document to the CC, the *Common Methodology for IT Security Evaluation* (CEM) [CEM12], defines the minimum set of assurance activities to conduct a CC evaluation, using the criteria and evaluation evidence defined in the CC. The original CC/CEM approach is basically a monolithic evaluation scheme.

However, the monolithic approach does not scale well because the evaluation effort grows exponentially with the complexity of the evaluation target. Moreover, if only a single subcomponent of an evaluated system is changed, the entire evaluation result is completely invalidated because in a monolithic approach, the ramifications of such changes for the assurance argument cannot be confined satisfactorily.

To escape these drawbacks and to keep pace with the growing assurance demands, a more incremental approach is called for, that is, an evaluation scheme whose effort grows only moderately with increasing system complexity and whose partial evaluation results can be reused for the evaluation of similar or slightly modified system configurations. To this end, so-called compositional evaluation approaches have been suggested.

1.4 Compositional Security Evaluation

In a compositional evaluation, the individual components of a system are evaluated independently, and these partial evaluation results are composed to derive the overall evaluation verdict with minimum additional effort. Ideally, if some part of the system configuration changes, only this part needs to be re-evaluated, while the compositional assurance argument for the remaining parts and for the overall system remains (mostly) intact. As a consequence, compositional evaluation offers potential for a fast and efficient certification of systems composed of previously certified sub components, and for a speedy re-certification of modified or updated systems.

Although composition is very attractive in terms of reduction in evaluation complexity (“divide et impera”) and evaluation effort, a sound composition of security arguments is far from trivial, as it raises some serious conceptual problems [DRDC04]. Note that in general, security is a system property that is neither invariant with respect to composition nor refinement. That is, assembling a system from secure parts need not yield a secure system, and refining a secure design into a more elaborated implementation may also compromise the security of the final product. These observations show that the composition of security evaluation results requires careful consideration and some additional integration effort.

1.5 Scope of this Report

To overcome these fundamental limitations of compositional security, various composition approaches have been suggested. In Chapter 2, we introduce two of the most renowned and most influential composition schemes for security evaluations, and we contrast their relative merits and drawbacks. Considering the specific needs of the avionics domain, the more promising of the two approaches is then surveyed in closer detail in Chapter 3.

Finally, Chapter 4 presents a small case study in the context of an avionic evaluation problem posed by EADS. The case study illustrates the application of the proposed evaluation scheme and scrutinizes its suitability for this particular problem domain.

Chapter 2

Compositional Evaluation Approaches

After the publication and widespread application of the Common Criteria, it was found that the CC evaluation effort often scales poorly for complex evaluation targets and that even marginal modifications of a CC-certified product annihilate the validity of the certificate, necessitating a full-scale re-evaluation of the modified product [DRDC04]. To address these (and other) problems, the Common Criteria Maintenance Board (CCMB) decided to include compositional elements into the CC evaluation framework.

2.1 Composed Assurance Packages (CAP)

In Version 3.1 of the CC [ISO15408, CC12], the CCMB added the so-called Composed Assurance Packages (CAP-A, -B, and -C) as an alternative for the non-compositional Evaluation Assurance Level (EAL1 – EAL7) packages. CAPs aim at evaluation scenarios according to Figure 1, where a composed target of evaluation (TOE) is comprised solely of components that have already successfully passed CC evaluation in compliance with an EAL (or an equivalent) assurance package.

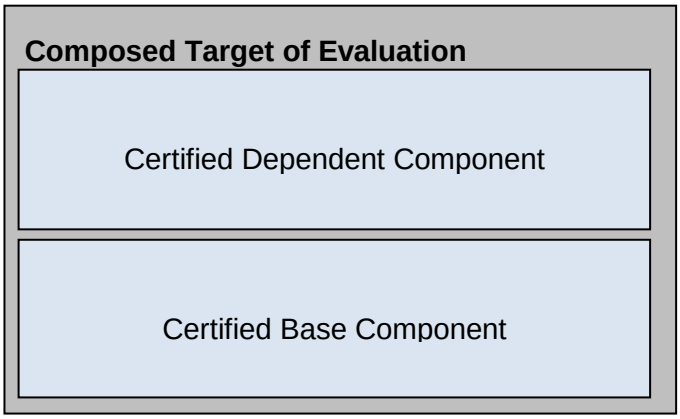

Figure 1: CAP evaluation scenario according to [CC12]

In support of the CAP approach, the CCMB introduced a new class of assurance requirements, Assurance Class Composition (ACO), whose components are mandatory for CAP-based evaluations (cf. [CC12], Part 3, Section 17). In addition to the ordinary CC assurance requirements, ACO requirements demand a composition rationale (ACO_COR). The composition rationale requires the evaluator to determine whether the appropriate assurance measures have been applied to each base component, and whether the base component is being used in its proper CC-evaluated configuration. Also, a detailed specification of the reliance of the dependent component(s) on the base component(s) is required (ACO_REL), as well as a vulnerability analysis targeted towards specific composition vulnerabilities (ACO_VUL) and an explicit composition testing (ACO_CTT). Furthermore, the evaluator must determine that the interface description provided for the base component is consistent with the reliance needs of the respective dependent component and provides sufficient evidence that the base component actually meets all expectations of the dependent component (ACO_DEV).

Besides these new composition-specific requirements, the CAP approach relies on the standard EAL assurance requirements by re-using existing assurance evidence obtained during a preceding EAL evaluation of the individual components. In essence, the CAP approach assumes that certified sub components are basically secure, and that potential vulnerabilities of the composed system only arise from component interaction at the interfaces between dependent and relied-upon components. Based on this assumption, it is sufficient to analyze and test the security and compatibility of the respective interfaces. Thus, undue duplication of effort can be avoided, and existing (monolithic) CC component certificates can be leveraged to speed-up the CAP certification of the composed system.

Apparently, the CAP evaluation scheme assumes that the TOE is only composed of certified components, with negligible additional "wiring" (hardware) or "glue code" (software) required for the integration of base and dependent component(s). If it were necessary to add extensive infrastructure to enable component interplay and to obtain the composed system, then this supplementary equipment would introduce significant additional vulnerability potential, thereby infringing the foundations of the CAP approach. Thus, genuine CAP evaluations are only admissible for pure compositions of certified sub components.

The CAP approach requires that each component of the composed TOE has been subject to individual EAL evaluation commensurate with the desired CAP level A, B, or C. For CAP-C ("methodically composed, tested, and reviewed"), for example, EAL4 ("methodically designed, tested, and reviewed") or higher is the adequate assurance level at which base component(s) and dependable component(s) need to be evaluated. Moreover, for an evaluation according to CAP the composed TOE requires at least EAL1 basic assurance. Note, though, that many of the EAL1 assurance requirements are already covered by related clauses of the ACO class. Consequently, EAL1 evaluation adds only little overhead to a CAP-C-based evaluation of a composed TOE.

However, despite its claimed reduction in evaluation effort, a CAP assessment may generate considerable work for the evaluator. For example, a CAP-C evaluation requires at least

- an individual EAL4 evaluation of each base component and each dependent component of the composed system;
- a basic EAL1 evaluation of the composed system;
- a final CAP-C evaluation of the composed system, including an ACO assessment *for each pair* of base and dependent components.[1]

The latter item, in particular, may cause a "combinatorial explosion" if multiple base and dependent components are involved.

Another drawback is that CAPs only consider resistance against an attacker with an attack potential up to "enhanced basic". This is due to the level of design information that can be provided through ACO_DEV, affecting the rigor of vulnerability analysis that can be performed by the evaluator. Therefore, the level of assurance arising from composed TOE evaluations using CAPs is limited to a level similar to that obtained from EAL4 evaluations[2], although the confidence in the individual components within the composed TOE may be much higher.

[1] If there are multiple base or dependent components, their mutual interference must also be considered in principle. However, the CAP approach is based on the implicit assumption that component interaction is essentially confined to interface functions, so it should be covered already by the pair wise assessment.

[2] In the strict sense, CAP levels and EAL ratings are incommensurable. However, CAP-C and EAL4 are reasonably similar in their requirements and their expected level of assurance to justify this comparison.

Unfortunately, CAP-C/EAL4 might provide insufficient security assurance for the safety attestation of highly safety-critical systems. For example, if safety according to Design Assurance Level (DAL) A [DO254, DO178C] is required, security confidence at EAL5 is probably a better match. Therefore, the CAP approach has limited applicability in the avionics domain.

2.2 CCDB Composite Product Evaluation (CPE)

The drawbacks of CAP evaluations also became apparent in other safety-sensitive domains, such as the smartcard business, where CC certification is the rule and high assurance is the norm. To better serve the needs of the smartcard domain, the Common Criteria Development Board (CCDB) issued the Common Criteria Supporting Document *Composite Product Evaluation for Smart Cards and Similar Devices* (CPE) [CCDB12]. The approach delineated in this document offers a more powerful and flexible alternative to the inherently limited CAP evaluation scheme.

In fact, the CPE approach does not address smartcards only, but any other security IT technology where an independently evaluated product is part of a final composite product to be evaluated. Figure 2 shows the underlying evaluation scenario.

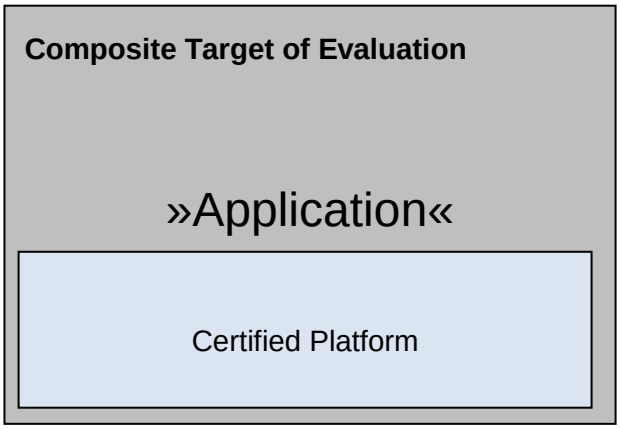

Figure 2: CPE evaluation scenario according to [CCDB12]

The CPE approach assumes a composite TOE consisting of a “platform” component (the equivalent of a CAP “base component”) already certified and an enclosing, still uncertified composite—often called “application” (the equivalent of a CAP “dependent component”)—using this platform, for which overall certification is sought. In contrast to the CAP scheme (cf. Figure 1, p. 3), CPE tolerates composite products comprising an arbitrary amount of encircling “wiring” in addition to one or more certified sub components.

Moreover, in the CPE model the platform component is not required to offer only “strictly functional” properties related to security: The platform is assumed to provide mechanisms to protect the composite product assets, but the composite product behavior generally depends widely on the software application having to use, to configure, and to activate these security mechanisms. The CAP approach, on the other hand, assumes that the interaction between the sub components of a composed TOE is confined to explicit interface calls.

For example, CPE can easily handle separation kernel platforms or integrated modular avionics (IMA) platforms providing partitioning as their main contribution to securing the composite TOE, which can hardly be mapped to a few discrete “security functions”. With the CAP approach, which critically depends on a clean specification of the interfaces between base and dependent components, it is more difficult to delineate the precise “functional interface” such a partitioning platform is providing to its dependent application components.

To enable such a flexible evaluation approach, the CC Security Target specification and supporting documentation of the platform component are supposed to be taken into account early in the specification and the development of the enclosing application of the composite product. For example, all obligations and restrictions documented in the platform's user guidance (AGD_OPE) and deployment guidance (AGD_PRE) are considered when integrating the platform component into the overall system architecture. In contrast, CAP components can be developed independently, and only on system integration is their mutual interaction considered. As a consequence, CPE requires a more coordinated integration of certified components into a composite product's security architecture, whereas the CAP approach—if applicable—enables "late integration" strategies. Support for late integration is a convenient property when some component of a composite product (e.g., the separation kernel platform of an IMA architecture) needs to be replaced during or after the composite evaluation.

Another crucial difference between the CAP and the CPE scheme is that the former restricts the security assurance that can be obtained to an assurance level of CAP-C (comparable to EAL4) and to resistance against "enhanced basic" attacks, whereas CPE does not limit the composite evaluation in EAL or in resistance against attacks.

Thus, CPE is preferable to CAP if

- the security-related properties of the certified sub components are known early in the design of the composite TOE

  and

- standard EAL compliance (as opposed to CAP compliance) is sought for the composite TOE,
- high assurance levels (e.g., EAL5–7 with respect to security or security-dependent DAL A–B with respect to safety[3]) are required, or
- certified sub components need to be embedded into composite TOEs comprising uncertified sub components as well.

Many avionics security and safety evaluations match these characteristics quite well. Therefore, we selected CPE as the method of choice for tackling the evaluation problem posed by EADS (see Chapter 4) that triggered this report.

[3] Even for safety assurance level DAL A, a security assurance level of EAL5 is generally considered as sufficient assurance for the airworthiness with respect to security threats in typical application scenarios (see, e.g., the draft version of the RTCA/EUROCAE airworthiness standard ED-203 [ED203]). That is, from a safety-perspective, EAL6 or even EAL7 are rarely required.

Chapter 3

Survey of the Composite Product Evaluation Approach

One of CPE's main objectives is to enable installing one or several applications onto an already certified platform in order to reduce the evaluation effort while keeping a high level of confidence. To this end, CPE provides rules and guidance for a transfer of knowledge between platform and application supplier and for a reuse of existing evaluation evidence.

In this chapter, we present an overview of the CPE approach, and we look into the assurance activities prescribed by it.

3.1 Overview

According to [CCDB12], a composite evaluation comprises the following fundamental steps:

- First, the platform component must be evaluated at an assurance level commensurate with the desired overall assurance of the composite evaluation, such as EAL5.
- Second, the composite TOE is evaluated according to some assurance package, such as EAL5, augmented with some additional "composite security assurance requirements" as defined in [CCDB12]. This composite evaluation rests upon the results of the preceding platform evaluation, reusing the earlier findings.

Note that in CPE, there is no separate evaluation of the surrounding application component (as would be required with the CAP approach), but composite and underlying platform are evaluated together as an integrated system in one sweep.

Apart from defining a composite assurance requirements package, the CPE framework [CCDB12, CCDB12b, and CCDB12c] also includes a corresponding composite evaluation method complementing the original CC evaluation method [CEM12]: For each of its composite assurance requirements, CPE defines appropriate evaluator actions (so-called "work units" in CPE parlance) for their proper evaluation. In concert with the composite assurance requirements, these CPE work units augment the standard CEM actions with specific instructions for composite products. Thus, the CPE methodology defines composite-specific developer and evaluator action elements with a clear statement on the information needed from the platform developer, and it provides an agreed framework for information transfer from the platform evaluator to the composite product evaluator.

To enable efficient evaluation of the composite TOE, CPE's evaluation method avoids requiring additional work from the developer of the certified platform component, but refers only to information that is readily available from the platform evaluation tasks. Moreover, the evaluator of the composite product need not even know the detailed design of the platform (which might contain proprietary information of the platform supplier). Thus, the key elements of an efficient reuse of established platform evaluation results are:

- There is no need for details on the platform development assurance requirements class ADV.

- The user guidance (cf. AGD assurance requirements class) of the platform is considered early in the development of the composite product and provides all interfaces information needed.
- The evaluated interfaces of the platform are relied upon.
- Test (cf. ATE class) and vulnerability assessment (cf. AVA class) are performed on the composite product taking advantage of platform evaluation results.

Of course, all relevant interfaces between platform and application are within the scope of the composite product evaluation, but most of them have already been security assured during platform certification.

3.2 Reusability of Platform Evaluation Certificates

In order to reuse existing component certificates, the evaluation of these components must be sufficiently up-to-date. As a general rule, the composite product certification body will ask for a reassessment of the platform if the platform's *Evaluation Technical Report for Composition* (ETR) dates back more than one and a half year before the submission of the report containing the full results of the composition penetration tests (cf. [CCDB12], Section 6, Rule R39). This reassessment consists of either a re-evaluation of the platform focusing on a renewal of the vulnerability analysis (surveillance task) or alternatively, a confirmation statement of the platform certification body may be requested.

3.3 CPE Security Assurance Requirements Families

Unlike CAP, the CPE approach does not define any new assurance classes; rather, it augments existing CC assurance classes with composition-specific *_COMP families. These new families cover the following activities:

- Consistency check of the Security Target specifications
- Integration of the application in the configuration management system
- Consistency check of delivery and acceptance procedures
- Composite product design compliance check
- Composite product functional testing
- Composite product vulnerability assessment

We will describe each of these new assurance families below.

3.3.1 Consistency Check of the Security Target Specifications (ASE_COMP)

The aim of this assurance family is to determine whether the Security Target specification of the composite product does not contradict the Security Target of the underlying platform.

The ASE_COMP family augments the CC Version 3.1 assurance families ASE_OBJ, ASE_REQ, and ASE_SPD.

Table 1 shows the assurance elements of ASE_COMP. The corresponding assurance elements of the standard CC catalog for levels EAL4 and EAL5 are shown in the "Remarks" column of the table in blue font color for reference, as well as related CAP-C requirements. Furthermore, for each evaluation requirement the "Remarks" column lists the corresponding CPE evaluation work units

Assurance Class/Family/Element	Remarks
Security Target Evaluation	Corresponding EAL/CAP Requirements • **CAP-C:** ASE_CCL.1, ASE_ECD.1, ASE_INT.1, ASE_OBJ.2, ASE_REQ.2, ASE_SPD.1, ASE_TSS.1 • **EAL4:** —same as CAP-C— • **EAL5:** —same as EAL4—
Consistency of composite product ST (ASE_COMP)	The aim of this family is to determine whether the Security Target of the composite product does not contradict the Security Target of the underlying platform.
ASE_COMP.1.1D The developer shall provide a statement of compatibility between the composite Security Target and the platform Security Target. This statement can be provided within the composite product Security Target.	**CAP-C:** → ACO_REL.1-1/ACO_REL.2-1 The evaluator shall check the reliance information to determine that it describes the functionality of the base dependent hardware, firmware and/or software that is relied upon by the dependent component TSF.
ASE_COMP.1.1C The statement of compatibility shall describe the separation of the platform TSF into relevant platform TSF being used by the composite ST and others.	
ASE_COMP.1.2C The statement of compatibility between the composite Security Target and the platform Security Target shall show (e.g. in form of a mapping) that the Security Targets of the composite product and of the underlying platform match, i.e. that there is no conflict between security environments, security objectives, and security requirements of the composite Security Target and the platform Security Target. It can be provided by indicating of the concerned elements directly in the Security Target for the composite product followed by explanatory text, if necessary.	**CAP-C:** → ACO_REL.1-2/ACO_REL.2-2 The evaluator shall examine the reliance information to determine that it accurately reflects the objectives specified for the operational environment of the dependent component.
	Work units related to ASE_COMP.1.2C: **ASE_COMP.1-3** The evaluator shall check that the security assurance requirements of the composite evaluation represent a subset of the security assurance requirements of the underlying platform. **(→ ASE_REQ)** **ASE_COMP.1-4** The evaluator shall examine the statement of compatibility to determine that all performed operations on the relevant TOE security functional requirements of the platform are appropriate for the composite ST. **(→ ASE_REQ)** **ASE_COMP.1-5**

Assurance Class/Family/Element	Remarks
	The evaluator shall examine the statement of compatibility to determine that the relevant TOE security objectives of the platform ST are not contradictory to those of the composite ST. **(→ ASE_OBJ)** **ASE_COMP.1-6** The evaluator shall examine the statement of compatibility to determine that the relevant threats of the platform ST are not contradictory to those of the composite ST. **(→ ASE_SPD)** **ASE_COMP.1-7** The evaluator shall examine the statement of compatibility to determine that the relevant organizational security policies of the platform ST are not contradictory to those of the composite ST. **(→ ASE_SPD)** **ASE_COMP.1-8** The evaluator shall examine the statement of compatibility to determine that the relevant organizational security policies of the platform ST are not contradictory to the threats of the composite ST and vice versa. **(→ ASE_SPD)** **ASE_COMP.1-9** The evaluator shall examine the statement of compatibility to determine that the list of the assumptions of the platform ST being significant for the composite ST is complete and consistent for the current composite TOE. **(→ ASE_SPD)** **ASE_COMP.1-10** The evaluator shall examine the statement of compatibility to determine that the significant security objectives for the operational environments of the platform ST are not contradictory to those of the composite ST. **(→ ASE_OBJ)**
ASE_COMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	
	Work units related to ASE_COMP.1.1E: **ASE_COMP.1-1** The evaluator shall check that the statement of compatibility describes the separation of the platform TSF into relevant platform TSF being used by the composite ST and others. (→ ASE_REQ) **ASE_COMP.1-2** The evaluator shall examine the statement of compatibility to determine that the platform TSF being used by the composite ST is complete and consistent for the current composite TOE. (→ ASE_REQ)

Table 1: Assurance elements of CPE assurance family ASE_COMP

The table shows that ASE_COMP is mainly concerned with the consistency of the Security Problem Definitions (SPDs) of the platform and the composite TOE, and with the consistency of the security objectives derived from these SPDs. The composite product evaluator has to examine the composite and the platform Security Target for any conflicting objectives or assumptions.

To this end, it is recommended that in step 1, the composite product Security Target is formulated independently of the platform Security Target. In step 2, the relevant intersection between the two Security Targets in terms of TOE Security Functionality (TSF) is determined. Finally in step 3, it is determined under which conditions the composite product can trust in and rely on the platform TSF being used without a new examination.

If multiple platform components are integrated into the composite product, determining the respective intersection between each platform Security Target and the composite product Security Target can become a laborious task. Fortunately, according to [CCDB12], Section 4.1, Rule R1, this effort can be reduced if some or all platform components comply with a common Protection Profile: In this case, part of the required matching need only be conducted once for the Protection Profile, and then all complying platform components can share these results.

3.3.2 Application Integration & Compatibility of Delivery and Acceptance (ALC_COMP)

The aims of this family are to determine whether

- the correct version of the application is installed onto the correct version of the underlying platform, and whether
- the delivery procedures of platform and application developers are compatible with the acceptance procedure of the composite product integrator.

The ALC_COMP family augments the CC Version 3.1 assurance families ALC_CMS, ALC_DEL, and AGD_PRE, all dealing with life-cycle aspects. Table 2 shows the assurance elements of ALC_COMP.

The assurance work units for ALC_COMP contain nothing unexpected. They just reemphasize the importance of employing the platform component only according to the certified deployment scenario and with the correct deployment parameters as documented in AGD_PRE, taking into account the required assurance during component delivery. And, of course, if the composite product has stringent delivery requirements, we can hardly include a platform with less stringent delivery requirements.

Assurance Class/Family/Element	Remarks
Life-cycle Support	Corresponding EAL/CAP Requirements: • **CAP-C:** ALC_CMC.1, ALC_CMS.2 • **EAL4:** ALC_CMC.4, ALC_CMS.4, ALC_DEL.1, ALC_DVS.1, ALC_LCD.1, ALC_TAT.1 • **EAL5:** ALC_CMC.4, ALC_CMS.5, ALC_DEL.1, ALC_DVS.1, ALC_LCD.1, ALC_TAT.2
Integration of composition parts and compatibility of delivery and acceptance procedures (ALC_COMP)	The aims of this family are to determine whether • the correct version of the application is installed onto/into the correct version of the underlying platform, and whether • the delivery procedures of platform developers and application developers are compatible with the acceptance procedure of the composite product

Assurance Class/Family/Element	Remarks
	integrator.
ALC_COMP.1.1D The developer shall provide components configuration evidence (cf. item #7 in [CCDB12], Section 4.7, Table 1).	
ALC_COMP.1.2D The developer shall provide an evidence for delivery and acceptance compatibility (cf. item #8 in in [CCDB12], Section 4.7, Table 1).	
ALC_COMP.1.1C The components configuration evidence shall show that (i) the evaluated version of the application has been installed onto / embedded into the certified version of the underlying platform and that (ii) configuration parameters prescribed by the platform and application developers are actually being used by the composite product integrator.	
ALC_COMP.1.2C The evidence for delivery and acceptance compatibility shall show that the delivery procedures of the platform and application developers are compatible with the acceptance procedure of the composite product Integrator.	
ALC_COMP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	
	Work units related to ALC_COMP.1.1E: **ALC_COMP.1-1** The evaluator shall check the evidence that the evaluated version of the application has been installed onto / embedded into the correct, certified version of the underlying platform. **(→ ALC_CMS)** **ALC_COMP.1-2** The evaluator shall examine the evidence for using configuration parameters to determine that the composite product integrator uses the configuration parameters prescribed by the platform and application developers. **(→ AGD_PRE)**
ALC_COMP.1.2E The evaluator shall confirm that the evidence for delivery compatibility is	**Application Note:** If there are no delivery interfaces between the platform and application developers and the composite product integrator or the assurance package

Assurance Class/Family/Element	Remarks
complete, coherent, and internally consistent.	chosen does not contain the family ALC_DEL (e.g. EAL1), this work unit is not applicable. The result of this work unit shall be integrated to the result of ALC_DEL.1.1C/ ALC_DEL.1-1.
	Work units related to ALC_COMP.1.2E: **ALC_COMP.1-3** The evaluator shall examine the evidence for compatibility of delivery interfaces to determine that delivery procedures of the platform and application developers are compatible with the acceptance procedure of the composite product integrator. **(→ ALC_DEL)**

Table 2: Assurance elements of CPE assurance family ALC_COMP

3.3.3 Composite Product Design Compliance (ADV_COMP)

The aim of this family is to determine whether the requirements on the application, imposed by the underlying platform, are fulfilled in the composite product.

The ADV_COMP family augments the CC Version 3.1 assurance families ADV_ARC, ADV_INT, ADV_IMP, and ADV_TDS, which address aspects of secure development. Table 3 gives an overview of the ADV_COMP assurance elements

Assurance Class/Family/Element	Remarks
Development	Corresponding EAL/CAP Requirements: • **CAP-C:** ADV_FSP.1, ACO_DEV, ACO_REL, ACO_COR • **EAL4:** ADV_ARC.1, ADV_FSP.4, ADV_IMP.1, ADV_TDS.3 • **EAL5:** ADV_ARC.1, ADV_FSP.5, ADV_IMP.1, ADV_INT.2, ADV_TDS.4
Composite design compliance (ADV_COMP)	The aim of this family is to determine whether the requirements on the application, imposed by the underlying platform, are fulfilled in the composite product.
ADV_COMP.1.1D The developer shall provide a design compliance justification (cf. item #6 as well as items #3, #4, #5 in [CCDB12], Section 4.7, Table 1).	
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 application, imposed by the underlying platform, are fulfilled in the composite product.	

Assurance Class/Family/Element	Remarks
ADV_COMP.1.1E The evaluator shall confirm that the rationale for design compliance is complete, coherent, and internally consistent.	**Application Notes:** • If the assurance package chosen does not contain the families ADV_TDS, ADV_ARC, or ADV_IMP (e.g. EAL1), this work unit is not applicable. • If there are no requirements of the platform concerning the TSF internal structure or the assurance package chosen does not contain the family ADV_INT, this work unit is not applicable.
	Work units related to ADV_COMP.1.1E: **ADV_COMP.1-1** The evaluator shall examine the rationale for design compliance to determine that all applicable requirements on the application, imposed by the underlying platform, are fulfilled by the composite product. **(→ ADV_ARC, ADV_IMP, ADV_TDS)** **ADV_COMP.1-2** The evaluator shall check the TSF internals of the composite TOE to determine that they do not contradict any design requirement imposed by the underlying platform. **(→ ADV_INT)**

Table 3: Assurance elements of CPE assurance family ADV_COMP

ADV_COMP ensures that the composite product employs the platform component in an appropriate application context. That is, the environment provided by the composite complies with the needs and assumptions of the platform component.

Note the difference between CPE and CAP: In CAP, we have to consider the *mutual* requirements of base component and dependent component on their respective peer: the dependent component must be suitable for the base component's needs, and vice versa. In CPE, only the former requirements needs to be assessed here, whereas the latter aspect is considered later during the evaluation of the overall composite product (which is more rigorous in CPE than in CAP, where the composed system is evaluated only superficially according to the less demanding CAP assurance requirements).

In contrast to CAP, CPE acknowledges the fact that the interface between the underlying platform and the application is an internal one; hence, a functional specification (ADV_FSP) at representation level is not sufficient for analyzing the design compliance. Instead, we also have to take into account platform services that do not provide a clean functional interface, such as domain separation, self-protection, or non-bypassability, which require an assessment at the level of architecture. The corresponding information may be found in the ADV_ARC-related documentation of the platform component.

3.3.4 Composite Product Functional Testing (ATE_COMP)

The aims of this family is to determine whether

- the test specifications are adequate and
- the composite product *as a whole* exhibits the properties necessary to satisfy the functional requirements of its Security Target.

The ATE_COMP family augments the CC Version 3.1 assurance families ATE_COV, ATE_FUN, and ATE_IND, all related to security aspects of testing. Table 4 shows the assurance elements of the ATE_COMP family.

ATE_COMP is exclusively concerned with integration testing, that is, testing the composite product fully assembled, including the platform module, without replacing individual components by emulation. It is assumed that both the platform and the application have already passed individual testing according to the canonical ATE assurance class.

More specifically, ATE_COMP requires the evaluator to analyze for each Security Functional Requirement (SFR) whether it directly depends on security properties of the platform and of the application. The evaluator shall verify that the integration tests performed by the developer cover at least all such SFRs.

Assurance Class/Family/Element	Remarks
Tests	Corresponding EAL/CAP Requirements • **CAP-C:** ATE_IND.2, ACO_CTT.2 • **EAL4:** ATE_COV.2, ATE_DPT.1, ATE_FUN.1, ATE_IND.2 • **EAL5:** ATE_COV.2, ATE_DPT.3, ATE_FUN.1, ATE_IND.2
Composite functional testing (ATE_COMP)	The aim of this family is to determine whether the test specifications are adequate and whether the composite product as a whole exhibits the properties necessary to satisfy the functional requirements of its Security Target.
ATE_COMP.1.1D The developer shall provide a set of tests as required by the assurance package chosen.	**CAP-C:** → 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.
ATE_COMP.1.2D The developer shall provide the composite TOE for testing.	**CAP-C:** → ACO_CTT.2.3D The developer shall provide the composed TOE for testing.
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 TOE provided shall be suitable for testing.	**CAP-C:** → ACO_CTT.2.4C The base component shall be suitable for testing.
ATE_COMP.1.1E	**Application Notes:**

Assurance Class/Family/Element	Remarks
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	• If the assurance package chosen does not contain the families ATE_FUN and ATE_COV (e.g., EAL1), this work unit is not applicable. • In order to perform this work unit, the evaluator shall analyze, for each TSF, whether it directly depends on security properties of the platform and of the application. Then the evaluator shall verify that the integration tests performed by the developer cover at least all such TSF.
	Work units related to ATE_COMP.1.1E: **ATE_COMP.1-1** The evaluator shall examine that the developer performed the integration tests for all SFRs having to be tested on the composite product as a whole. **(→ ATE_COV, ATE_FUN)**
ATE_COMP.1.2E The evaluator shall specify, perform and document a set of own integration tests to confirm that the composite TOE operates as specified.	**CAP-C:** → 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. For this assurance element, the evaluator determines the share of the platform part of the TOE in enforcing of the composite ST. Next, the evaluator checks, for each such composite SFR, whether the platform's share has been covered by the platform certificate. Finally, the evaluator refers to the "ETR for Composition" and checks for any explicit requirements for performing tests in the context of the composite evaluation.
	Work units related to ATE_COMP.1.2E: **ATE_COMP.1-2** The evaluator shall determine the minimal amount of the integration tests being necessary for the current composite evaluation. **(→ ATE_COV)** **ATE_COMP.1-3** The evaluator shall perform the standard evaluator actions in the context of the assurance family ATE_IND on the set of the integration tests using the composite product as a whole. **(→ ATE_IND)**

Table 4: Assurance elements of CPE assurance family ATE_COMP

As stated in Table 4, the ATE_COMP family provides little information on how to reuse the evaluation results of the platform certification in order to reduce the testing effort during composite product evaluation. However, [CCDB12] offers additional guidance on how to restrict integration testing essentially on those platform dependencies that have not received sufficient attention during platform attestation (typically, because the respective platform functionality was originally not considered TSF-relevant for the platform, but is now critical for the composite TSF). Since the amount, the coverage, and the depth of the functional tests of

the platform have already been validated by the platform certificate, it is not necessary to re-perform these tasks in the composite evaluation during integration testing (cf. [CCDB12], Section 4.5, Rule R12).

3.3.5 Composite Product Vulnerability Assessment (AVA_COMP)

The aim of this family is to determine the exploitability of flaws or weaknesses in the composite TOE *as a whole* in the intended environment.

The AVA_COMP family augments the CC Version 3.1 assurance family AVA_VAN, which is concerned with vulnerability analysis. More specifically, AVA_COMP activities aim to refine AVA_VAN.1.3E (or the equivalent higher components if a higher assurance level is selected) and its canonical evaluation work units AVA_VAN.1-5 to AVA_VAN.1-8 [CEM12], all referring to penetration testing. Table 5 summarizes the assurance elements of AVA_COMP.

The suitability and sufficiency of AVA_COMP depends strongly on the correct composition of platform and surrounding application because the platform certificate remains only valid if the composition meets all of the platform's environment requirements. Otherwise, we cannot assume that the composing of the platform and the application will not create additional vulnerabilities inside the platform component. Accordingly, reusing and relying on prior platform evaluation results requires that the correctness-related composition activities—ASE_COMP.1, ALC_COMP.1, ADV_COMP.1, and ATE_COMP.1—are finalized with the verdict PASS and that the certificate for the platform covers all security properties needed for the composite product.

In fact, if the evaluator determined that composing of the platform and the application creates additional vulnerabilities *of the platform*, a contradiction to the verdict PASS for the correctness activities has to be supposed, or the certificate for the platform does not cover all security properties needed for the current composite product.

Assurance Class/Family/Element	Remarks
Vulnerability Assessment	Corresponding EAL/CAP Requirements • **CAP-C:** AVA_VAN.1, ACO_VUL.3 • **EAL4:** AVA-VAN.3 • **EAL5:** AVA_VAN.4
Composite vulnerability assessment (AVA_COMP)	The aim of this family is to determine the exploitability of flaws or weaknesses in the composite TOE *as a whole* in the intended environment.
AVA_COMP.1.1D The developer shall provide the composite TOE for penetrating testing.	**CAP-C:** → ACO_VUL.3.1D The developer shall provide the composed TOE for testing.
AVA_COMP.1.1C The composite TOE provided shall be suitable for testing as a whole.	**CAP-C:** → ACO_VUL.3.1C The composed TOE shall be suitable for testing.
AVA_COMP.1.1E The evaluator shall conduct penetration testing of the composite product as a whole building on evaluator's own vulnerability analysis, to ensure that the vulnerabilities being relevant for the composite ST are not exploitable.	**CAP-C:** → 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.

Assurance Class/Family/Element	Remarks
	Work units related to AVA_COMP.1.1E: **AVA_COMP.1-1** The evaluator shall examine the results of the vulnerability assessment for the underlying platform to determine that they can be reused for the composite evaluation. **(→ AVA_VAN)** **AVA_COMP.1-2** The evaluator shall specify, conduct and document penetration testing of the composite product *as a whole*, using the standard approach of the assurance family AVA_VAN. **(→ AVA_VAN)**

Table 5: Assurance elements of CPE assurance family AVA_COMP

If all conditions for correct composition are met, AVA_COMP reduces the evaluation effort insofar as it allows the evaluator to assume the correctness of all platform functionality without repeating any platform penetration tests. The evaluator must, however, perform dedicated penetration tests of the composite product, and part of the effort saved in AVA_COMP has simply been shifted to ADV_COMP and ATE_COMP.

Chapter 4

Applying CPE to an Avionic Security Gateway

This chapter evaluates the EURO-MILS avionic security gateway as a case study to illustrate the implications, advantages, and drawbacks of the CPE approach. This case study pursues two main objectives: The first is to explore compositional security evaluation approaches in general. The second goal is to devise representative key artifacts required for compositional CC certification, such as component Security Target specifications and their accompanying documents.

Due to the sensibility of the data being discussed in this chapter, the study is put into Chapter 1 of the **confidential appendix** to this document.

Chapter 5

Applying CPE to an Automotive MILS Infotainment Device

This chapter evaluates the EURO-MILS Automotive MILS Infotainment Device as a case study to illustrate the implications, advantages, and drawbacks of the CPE approach. The goal is to devise representative key artifacts required for compositional CC certification, such as component Security Target specifications and their accompanying documents.

Due to the sensibility of the data being discussed in this chapter, the study is put into Chapter 2 of the **confidential appendix** to this document.

Chapter 6

Summary and Outlook

This report details the assurance requirements and activities implied by the CPE evaluation approach, and it compares them to the corresponding requirements of the CAP-based approach for composite product evaluations. As we have seen, both evaluation methods have their merits and drawbacks when applied to TOEs such as the avionic security gateway. Altogether, from the two currently available methodologies for compositional certification in the framework of Common Criteria the CPE approach still seamed to be a reasonable choice for the compositional evaluation problem of our case study.

However, during the study and discussion with our project partners (in particular T-Systems and Thales), we identified some issues that required careful consideration. Among others, we reconfirmed the observation made in other projects that CC terminology is not well suited for specifying partitioning properties, which are central to modern IMA or MILS architectures and which play a key role in our modular ASG design. Part of the problem arises because the CC's distinction between user data and TSF data is inappropriate for certain types of partitioned evaluation targets. This can lead to awkward specifications. By using a compositional approach, we hope to shift much of this burden to the evaluation of the partitioning platform component, where the problem can be addressed once and for all so that applications on top of this platform can then just safely assume that correct partitioning is guaranteed.

Another issue on CPE is the composition of an evaluated base TOE with an unevaluated dependent component. Future compositional certification may want to re-use their already evaluated dependent components. For example this gets interesting by update circles of the base component to improved version or by exchange of dependent components. In the gateway use case this exchange could be an introduction of a new filter into the filter chain or also an update of one of the gateway function implementing partitions.

In its current set of methodologies the Common Criteria does not support the scenario of creating a composed system from evaluated (dependent and base) components with a high level of assurance, i.e. beyond EAL4. However, MILS as architecture enables system designs targeting high assurance by isolating component into partitions. Using the MILS principals and a separation kernel with sufficient properties should allow such a compositional certification and assurance scenario. Since both available methodologies of the CC, CPE and CAP, do not fit sufficiently to the MILS compositional environment, we started to propose a new evaluation methodology based on the non-interference criteria. The methodology is introduced as "Non-Interfering Composed Certification" (cf. EURO-MILS WP 3).

Chapter 7

List of Abbreviations

ASG	Avionic Security Gateway
CAP	Composed Assurance Package
CC	Common Criteria for Information Technology Security Evaluation
CCMB	Common Criteria Maintenance Board
CCDB	Common Criteria Development Board
CEM	Common Evaluation Methodology for IT Security
CPE	Composite Product Evaluation according to [CCDB12]
CPU	Central Processing Unit
DAL	Design Assurance Level
EADS	European Aeronautic Defence and Space Group
EAL	Evaluation Assurance Level
ETR	Evaluation Technical Report
IMA	Integrated Modular Avionics
IT	Information Technology
MILS	Multiple Independent Levels of Security
PIFP	Partitioned Information Flow Policy
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SKPP	Separation Kernel Protection Profile
SPD	Security Problem Definition
ST	Security Target specification
TSF	TOE Security Functionality
TOE	Target of Evaluation
WCET	Worst-case Execution Time

References and Bibliography

[AHO+06] Jim Alves-Foss, W. Scott Harrison, Paul Oman, and Carol Taylor (2006): The MILS Architecture for High-Assurance Embedded Systems. International Journal of Embedded Systems, Vol. 2, No. 3/4, pp. 239–247 http://www.researchgate.net/publication/220309643_The_MILS_architecture_for_high-assurance_embedded_systems/file/d912f50fee695f0273.pdf

[ARINC811] Airlines Electronic Engineering Committee (2005): Commercial Aircraft Information Security Concepts of Operation and Process Framework. ARINC Report 811

[CC12] Common Criteria Maintenance Board (2012): Common Criteria for Information Technology Security Evaluation, CCv3.1 Revision 4 (CCMB-2012-09-001, -002, -003) http://www.commoncriteriaportal.org/cc/

[CCDB12] Common Criteria Development Board (2012): Composite Product Evaluation for Smart Cards and Similar Devices. Common Criteria Supporting Document — Mandatory Technical Document, Version 1.2 (CCDB-2012-04-001) http://www.commoncriteriaportal.org/files/supdocs/CCDB-2012-04-001.pdf

[CCDB12b] Common Criteria Development Board (2012): Security Architecture Requirements (ADV_ARC) for Smart Cards and Similar Devices. Supporting Document — Guidance, Version 2.0 (CCMB-2012-04-03) http://www.commoncriteriaportal.org/files/supdocs/CCDB-2012-04-003.pdf

[CCDB12c] Common Criteria Development Board (2012): Security Architecture Requirements (ADV_ARC) for Smart Cards and Similar Devices — Appendix 1.Supporting Document — Guidance, Version 2.0 (CCMB-2012-04-04) http://www.commoncriteriaportal.org/files/supdocs/CCDB-2012-04-004.pdf

[CEM12] Common Criteria Maintenance Board (2012): Common Methodology for Information Technology Security Evaluation, CEMv3.1 Revision 4 (CCMB-2012-09-004) http://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R4.pdf

[DO178C] RTCA/EUROCAE (2012): Software Considerations in Airborne Systems and Equipment Certification. DO-178C / ED-12C http://www.rtca.org/store_list.asp

[DO254] RTCA/EUROCAE (2000): Design Assurance Guidance for Airborne Electronic Hardware. DO-254 / ED-80 http://www.rtca.org/store_list.asp

[DRDC04] Defence R&D Canada (2004): Review of the Composability Problem for System Evaluation. DRDC Ottawa CR 2004-19 http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.1268

[EM:D11.1] EURO-MILS consortium: Deliverable D11.1: Project Requirements: Classification, Cross-domain analysis and High-Level Architecture

[EM:D12.3] EURO-MILS consortium: Deliverable D12.3: Multiple Independent Levels of Security: Operating System (MILS PP: Operating System)

[EM:D21.1] EURO-MILS consortium: Deliverable D21.1: MILS Architecture

[ED203] RTCA/EUROCAE(2011): Airworthiness security methods and considerations. DO-YY3/ED-203, Working Draft

[IAD07] Information Assurance Directorate (2007): U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness, Version 1.03
http://www.niap-ccevs.org/pp/pp_skpp_hr_v1.03.pdf

[ISO15408] ISO/IEC 15408:2009: Information technology — Security techniques — Evaluation criteria for IT security (= CCv3.1)

[ISO18045] ISO/IEC 18045:2008: Information technology — Security techniques — Methodology for IT security evaluation (= CEMv3.1)

[Pre12] Christopher Preschern (2012): Catalog of Security Tactics linked to Common Criteria Requirements. In: Proc. 19th Conference on Pattern Languages of Programs (PLoP'12), October 19–21, Tucson, Arizona
http://www.hillside.net/plop/2012/index.php?nav=program#acceptedpapers

[VBC+05] W. Vanfleet, R. Beckwith, B. Calloni, J. Luke, C. Taylor, and G. Uchenick (2005): MILS: Architecture for High-Assurance Embedded Computing. CrossTalk: Journal of Defence Software Engineering, Vol. 18, No. 8, pp. 12–16
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.170.4270&rep=rep1&type=pdf