Systeme hoher Qualität und Sicherheit
Universität Bremen, WS 2013/14

Lecture 02 (28.10.2013)

Concepts of Safety and Security

Christoph Lüth
Christian Liguda

Where are we?

Synopsis

- **If** you want to write safety-criticial software, **then** you need to adhere to state-of-the-art practise **as** encoded by the relevant norms & standards.
- Today:
  - What is safety and security?
  - Why do we need it? Legal background.
  - How is it ensured? Norms and standards
    - IEC 61508 – Functional safety
    - IEC 15408 – Common criteria (security)

The Relevant Question

- If something goes wrong:
  - Whose fault is it?
  - Who pays for it?
- That is why most (if not all) of these standards put a lot of emphasis on process and traceability. Who decided to do what, why, and how?
- The **bad** news:
  - As a qualified professional, you may become personally liable if you deliberately and intentionally (*grob vorsätzlich*) disregard the state of the art.
- The **good** news:
  - Pay attention here and you will be sorted.

Safety: IEC 61508 and other norms & standards

What is Safety?

- Absolute definition:
  - „Safety is freedom from accidents or losses."
    - Nancy Leveson, „Safeware: System safety and computers"
- But is there such a thing as absolute safety?
- Technical definition:
  - „Sicherheit: Freiheit von unvertretbaren Risiken"
    - IEC 61508-4:2001, §3.1.8
- Next week: a safety-critical development process

Some Terminology

- Fail-safe vs. Fail operational
- Safety-critical, safety-relevant (*sicherheitskritisch*)
  - General term -- failure may lead to risk
- Safety function (*Sicherheitsfunktion*)
  - Techncal term, that functionality which ensures safety
- Safety-related (*sicherheitsgerichtet, sicherheitsbezogen*)
  - Technical term, directly related to the safety function

Legal Grounds

- The machinery directive:
  The Directive 2006/42/EC of the European Parliament and of the Council of 17 May 2006 on machinery, and amending Directive 95/16/EC (recast)
- Scope:
  - Machineries (with a drive system and movable parts).
- Structure:
  - Sequence of whereas clauses (explanatory)
  - followed by 29 articles (main body)
  - and 12 subsequent annexes (detailed information about particular fields, e.g. health & safety)
- Some application areas have their own regulations:
  - Cars and motorcycles, railways, planes, nuclear plants ...

What does that mean?

- Relevant for **all** machinery (from tin-opener to AGV)
- **Annex IV** lists machinery where safety is a concern
- Standards encode current best practice.
  - Harmonised standard available?
- External certification or self-certification
  - Certification ensures and documents conformity to standard.
- **Result:** CE
- Note that the scope of the directive is market harmonisation, not safety – that is more or less a byproduct.

The Norms and Standards Landscape

- First-tier standards (*A-Normen*):
  - General, widely applicable, no specific area of application
  - Example: IEC 61508
- Second-tier standards (*B-Normen*):
  - Restriction to a particular area of application
  - Example: ISO 26262 (IEC 61508 for automotive)
- Third-tier standards (*C-Normen*):
  - Specific pieces of equipment
  - Example: IEC 61496-3 ("*Berührungslos wirkende Schutzeinrichtungen*")
- Always use most specific norm.

Norms for the Working Programmer

- IEC 61508:
  - *"Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems (E/E/PE, or E/E/PES)"*
  - Widely applicable, general, considered hard to understand
- ISO 26262
  - Specialisation of 61508 to cars (automotive industry)
- DIN EN 50128
  - Specialisation of 61508 to software for railway industry
- RTCA DO 178-B:
  - *"Software Considerations in Airborne Systems and Equipment Certification"*
  - Airplanes, NASA/ESA
- ISO 15408:
  - *"Common Criteria for Information Technology Security Evaluation"*
  - Security, evolved from TCSEC (US), ITSEC (EU), CTCPEC (Canada)

Introducing IEC 61508

- Part 1: Functional safety management, competence, **establishing SIL targets**
- Part 2: Organising and managing the life cycle
- Part 3: **Software requirements**
- Part 4: Definitions and abbreviations
- Part 5: Examples of methods for the determination of safety-integrity levels
- Part 6: Guidelines for the application
- Part 7: Overview of techniques and measures

How does this work?

1. Risk analysis determines the safety integrity level (SIL)
2. A hazard analysis leads to safety requirement specification.
3. Safety requirements must be satisfied
   - Need to verify this is achieved.
   - SIL determines amount of testing/proving etc.
4. Life-cycle needs to be managed and organised
   - Planning: verification & validation plan
   - Note: personnel needs to be qualified.
5. All of this needs to be independently assessed.
   - SIL determines independence of assessment body.

Safety Integrity Levels

SIL	High Demand (more than once a year)	Low Demand (once a year or less)
4	10^{-9} < P/hr < 10^{-8}	10^{-5} < P/yr < 10^{-4}
3	10^{-8} < P/hr < 10^{-7}	10^{-4} < P/yr < 10^{-3}
2	10^{-7} < P/hr < 10^{-6}	10^{-3} < P/yr < 10^{-2}
1	10^{-6} < P/hr < 10^{-5}	10^{-2} < P/yr < 10^{-1}

- P: Probabilty of dangerous failure (per hour/year)
- Examples:
  - High demand: car brakes
  - Low demand: airbag control
- Which SIL to choose? → Risk analysis
- Note: SIL only meaningful for specific safety functions.

Establishing target SIL I

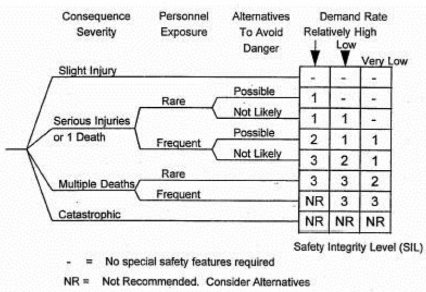
- IEC 61508 does not describe standard procedure to establish a SIL target, it allows for alternatives:
- Quantitative approach
  - Start with target risk level
  - Factor in fatality and frequency

Maximum tolerable risk of fatality	Individual risk (per annum)
Employee	10^{-4}
Public	10^{-5}
Broadly acceptable („Neglibile")	10^{-6}

- Example:
  - Safety system for a chemical plant
  - Max. tolerable risk exposure A=10^{-6}
  - B= 10^{-2} hazardous events lead to fatality
  - Unprotected process fails C= 1/5 years
  - Then Failure on Demand E = A/(B*C) = 5*10^{-3}, so SIL 2

Establishing target SIL II

- Risk graph approach



Example: safety braking system for an AGV

What does the SIL mean for the development process?

- In general:
  - „Competent“ personnel
  - Independent assessment („four eyes“)
- SIL 1:
  - Basic quality assurance (e.g ISO 9001)
- SIL 2:
  - Safety-directed quality assurance, more tests
- SIL 3:
  - Exhaustive testing, possibly formal methods
  - Assessment by separate department
- SIL 4:
  - State-of-the-art practices, formal methods
  - Assessment by separate organisation

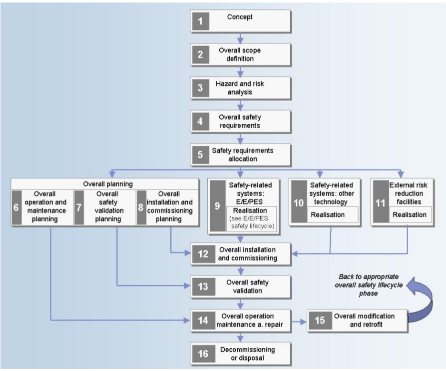
Increasing SIL by redudancy

- One can achieve a higher SIL by combining **independent** systems with lower SIL („*Mehrkanalsysteme*“).
- Given two systems A, B with failure probabilities P_A, P_B, the chance for failure of both is (with P_{CC} probablity of common-cause failures):

$$P_{AB} = P_{CC} + P_A P_B$$

- Hence, combining two SIL 3 systems may give you a SIL 4 system.
- However, be aware of **systematic** errors (and note that IEC 61508 considers all software errors to be systematic).
- Note also that for fail-operational systems you need three (not two) systems.

The Safety Life Cycle



The Software Development Process

- 61508 mandates a V-model software development process
  - More next lecture
- Appx A, B give normative guidance on measures to apply:
  - Error detection needs to be taken into account (e.g runtime assertions, error detection codes, dynamic supervision of data/control flow)
  - Use of strongly typed programming languages (see table)
  - Discouraged use of certain features: recursion(!), dynamic memory, unrestricted pointers, unconditional jumps
  - Certified tools and compilers must be used.
    - Or `proven in use´

Proven in Use

- As an alternative to systematic development, statistics about usage may be employed. This is particularly relevant
  - for development tools (compilers, verification tools etc),
  - and for re-used software (in particular, modules).
  - Note that the previous use needs to be to the same specification as intended use (eg. compiler: same target platform).

SIL	Zero Failure		One Failure	
1	12 ops	12 yrs	24 ops	24 yrs
2	120 ops	120 yrs	240 ops	240 yrs
3	1200 ops	1200 yrs	2400 ops	2400 yrs
4	12000 ops	12000 yrs	24000 ops	24000 yrs

Table A.2, Software Architecture

Tabelle A.2 – Softwareentwurf und Softwareentwicklung: Entwurf der Software-Architektur (siehe 7.4.3)

Verfahren/Maßnahme *	siehe	SIL1	SIL2	SIL3	SIL4
1 Fehlererkennung und Diagnose	C.3.1	o	+	++	++
2 Fehlererkennende und -korrigierende Codes	C.3.2	+	+	+	++
3a Plausibilitätskontrollen (Failure assertion programming)	C.3.3	+	+	+	++
3b Externe Überwachungseinrichtungen	C.3.4	o	+	+	+
3c Diversitäre Programmierung	C.3.5	+	+	+	++
3d Regenerationsblöcke	C.3.6	+	+	+	+
3e Rückwärtsregeneration	C.3.7	+	+	+	+
3f Vorwärtsregeneration	C.3.8	+	+	+	+
3g Regeneration durch Wiederholung	C.3.9	+	+	+	++
3h Aufzeichnung ausgeführter Abschnitte	C.3.10	o	+	+	++
4 Abgestufte Funktionseinschränkungen	C.3.11	+	+	++	++
5 Künstliche Intelligenz – Fehlerkorrektur	C.3.12	o	--	--	--
6 Dynamische Rekonfiguration	C.3.13	o	--	--	--
7a Strukturierte Methoden mit z. B. JSD, MAS-COT, SADT und Yourdon.	C.2.1	++	++	++	++
7b Semi-formale Methoden	Tabelle B.7	+	+	++	++
7c Formale Methoden z. B. CCS, CSP, HOL, LOTOS, OBJ, temporäre Logik, VDM und Z	C.2.4	o	+	+	++

Table A.4- Software Design & Development

Tabelle A.4 – Softwareentwurf und Softwareentwicklung: detaillierter Entwurf (siehe 7.4.5 and 7.4.6)
(Dies beinhaltet Software-Systementwurf, Entwurf der Softwaremodule und Codierung)

Verfahren/Maßnahme *	siehe	SIL1	SIL2	SIL3	SIL4
1a Strukturierte Methoden wie z. B. JSD, MAS-COT, SADT und Yourdon	C.2.1	++	++	++	++
1b Semi-formale Methoden	Tabelle B.7	+	++	++	++
1c Formale Methoden wie z. B. CCS, CSP, HOL, LOTOS, OBJ, temporäre Logik, VDM und Z	C.2.4	o	+	+	++
2 Rechnergestützte Entwurfswerkzeuge	B.3.5	+	+	++	++
3 Defensive Programmierung	C.2.5	o	+	++	++
4 Modularisierung	Tabelle B.9	++	++	++	++
5 Entwurfs- und Codierungs-Richtlinien	Tabelle B.1	+	++	++	++
6 Strukturierte Programmierung	C.2.7	++	++	++	++

Table A.9 – Software Verification

Tabelle A.9 – Software-Verifikation (siehe 7.9)

Verfahren/Maßnahme *	siehe	SIL1	SIL2	SIL3	SIL4
1 Formaler Beweis	C.5.13	o	+	+	++
2 Statistische Tests	C.5.1	o	+	+	++
3 Statische Analyse	B.6.4 Tabelle B.8	+	++	++	++
4 Dynamische Analyse und Test	B.6.5 Tabelle B.2	+	++	++	++
5 Software-Komplexitätsmetriken	C.5.14	+	+	+	+

Table B.1 – Coding Guidelines

- Table C.1, programming languages, mentions:
  - ADA, Modula-2, Pascal, FORTRAN 77, C, PL/M, Assembler, …
- Example for a guideline:
  - MISRA-C: 2004, Guidelines for the use of the C language in critical systems.

Tabelle B.1 – Entwurfs- und Codierungs-Richtlinien (Verweisungen aus Tabelle A.4)

Verfahren/Maßnahme *	siehe	SIL1	SIL2	SIL3	SIL4
1 Verwendung von Codierungs-Richtlinien	C.2.6.2	++	++	++	++
2 Keine dynamischen Objekte	C.2.6.3	+	++	++	++
3a Keine dynamischen Variablen	C.2.6.3	o	+	++	++
3b Online-Test der Erzeugung von dynamischen Variablen	C.2.6.4	o	+	++	++
4 Eingeschränkte Verwendung von Interrupts	C.2.6.5	+	+	++	++
5 Eingeschränkte Verwendung von Pointern	C.2.6.6	o	+	++	++
6 Eingeschränkte Verwendung von Rekursionen	C.2.6.7	o	+	++	++
7 Keine unbedingten Sprünge in Programmen in höherer Programmiersprache	C.2.6.2	+	++	++	++

ANMERKUNG 1 Die Maßnahmen 2 und 3a brauchen nicht angewendet zu werden, wenn ein Compiler verwendet wird, der sicherstellt, dass genügend Speicherplatz für alle dynamischen Variablen und Objekte vor der Laufzeit zugeteilt wird, oder der Laufzeittests zur korrekten Online-Zuweisung von Speicherplatz einfügt.

* Es müssen dem Sicherheits-Integritätslevel angemessene Verfahren/Maßnahmen ausgewählt werden. Alternative oder gleichwertige Verfahren/Maßnahmen sind durch einen Buchstaben hinter der Nummer gekennzeichnet. Es muss nur eine(s) der alternativen oder gleichwertigen Verfahren/Maßnahmen erfüllt werden.

Table B.5 - Modelling

Tabelle B.5 – Modellierung (Verweisung aus der Tabelle A.7)

Verfahren/Maßnahme *	siehe	SIL1	SIL2	SIL3	SIL4
1 Datenflussdiagramme	C.2.2	+	+	+	+
2 Zustandsübergangsdiagramme	B.2.3.2	o	+	++	++
3 Formale Methoden	C.2.4	o	+	+	++
4 Modellierung der Leistungsfähigkeit	C.5.20	+	++	++	++
5 Petri-Netze	B.2.3.3	o	+	++	++
6 Prototypenerstellung/Animation	C.5.17	+	+	+	+
7 Strukturdiagramme	C.2.3	+	+	+	++

ANMERKUNG Sollte eine spezielles Verfahren in dieser Tabelle nicht vorkommen, darf nicht angenommen werden, dass dieses nicht in Betracht gezogen werden darf. Es sollte zu dieser Norm in Einklang stehen.

* Es müssen dem Sicherheits-Integritätslevel angemessene Verfahren/Maßnahmen ausgewählt werden.

Certification

- Certiciation is the process of showing **conformance** to a **standard**.
- Conformance to IEC 61508 can be shown in two ways:
  - Either that an organisation (company) has in principle the ability to produce a product conforming to the standard,
  - Or that a specific product (or system design) conforms to the standard.
- Certification can be done by the developing company (self-certification), but is typically done by an **accredited** body.
  - In Germany, e.g. the TÜVs or the Berufsgenossenschaften (BGs)
- Also sometimes (eg. DO-178B) called `qualification'.

Security: The Common Criteria

Universität Bremen

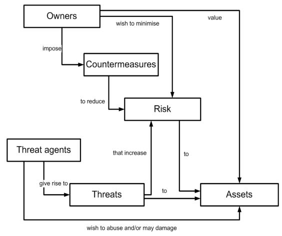
Common Criteria (IEC 15408)

- This multipart standard, the Common Criteria (CC), is meant to be used as the basis for evaluation of **security properties** of IT products and systems. By establishing such a common criteria base, the results of an IT security evaluation will be meaningful to a wider audience.
- The CC is useful as a guide for the development of products or systems with IT security functions and for the procurement of commercial products and systems with such functions.
- During evaluation, such an IT product or system is known as a **Target of Evaluation** (TOE) .
  - Such TOEs include, for example, operating systems, computer networks, distributed systems, and applications.

Common Criteria

General Model

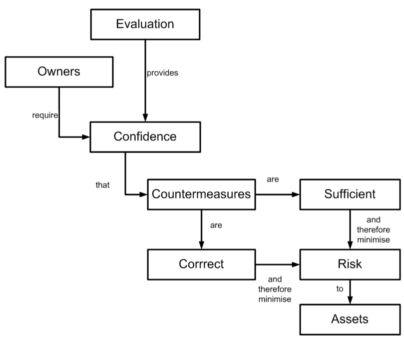
- Security is concerned with the protection of assets. Assets are entities that someone places value upon.
- Threats give rise to risks to the assets, based on the likelihood of a threat being realized and its impact on the assets
- (IT and non-IT) Counter-measures are imposed to reduce the risks to assets.



Common Criteria (CC)

- The CC addresses protection of information from unauthorized disclosure, modification, or loss of use. The categories of protection relating to these three types of failure of security are commonly called **confidentiality**, **integrity**, and **availability**, respectively.
- The CC may also be applicable to aspects of IT security outside of these three.
- The CC concentrates on **threats** to that information arising from human activities, whether malicious or otherwise, but may be applicable to some non-human threats as well.
- In addition, the CC may be applied in other areas of IT, but makes no claim of competence outside the strict domain of IT security.

Concept of Evaluation



Requirements Analysis

- The **security environment** includes all the laws, organizational security policies, customs, expertise and knowledge that are determined to be relevant.
  - It thus defines the context in which the TOE is intended to be used.
  - The security environment also includes the threats to security that are, or are held to be, present in the environment.
- A statement of applicable **organizational security policies** would identify relevant policies and rules.
  - For an IT system, such policies may be explicitly referenced, whereas for a general purpose IT product or product class, working assumptions about organizational security policy may need to be made.

Requirements Analysis

- A statement of **assumptions** which are to be met by the environment of the TOE in order for the TOE to be considered secure.
  - This statement can be accepted as axiomatic for the TOE evaluation.
- A statement of **threats** to security of the assets would identify all the threats perceived by the security analysis as relevant to the TOE.
  - The CC characterizes a threat in terms of a threat agent, a presumed attack method, any vulnerabilities that are the foundation for the attack, and identification of the asset under attack.
- An assessment of **risks** to security would qualify each threat with an assessment of the likelihood of such a threat developing into an actual attack, the likelihood of such an attack proving successful, and the consequences of any damage that may result.

Requirements Analysis

- The intent of determining **security objectives** is to address all of the security concerns and to declare which security aspects are either addressed directly by the TOE or by its environment.
  - This categorization is based on a process incorporating engineering judgment, security policy, economic factors and risk acceptance decisions.
  - Corresponds to (part of) requirements definition!
- The results of the analysis of the security environment could then be used to state the security objectives that counter the identified threats and address identified organizational security policies and assumptions.
- The security objectives should be consistent with the stated operational aim or product purpose of the TOE, and any knowledge about its physical environment.

Requirements Analysis

- The security objectives for the environment would be implemented within the IT domain, and by non-technical or procedural means.
- Only the security objectives for the TOE and its IT environment are addressed by IT security requirements.

Requirements Analysis

- The IT **security requirements** are the refinement of the security objectives into a set of security requirements for the TOE and security requirements for the environment which, if met, will ensure that the TOE can meet its security objectives.
- The CC presents security requirements under the distinct categories of functional requirements and assurance requirements.
- Functional requirements
  - Security behavior of IT-system
  - E.g. identification & authentication, cryptography,...
- Assurance Requirements
  - Establishing confidence in security functions
  - Correctness of implementation
  - E.g. Developement, life cycle support, testing, ...

Functional Requirement

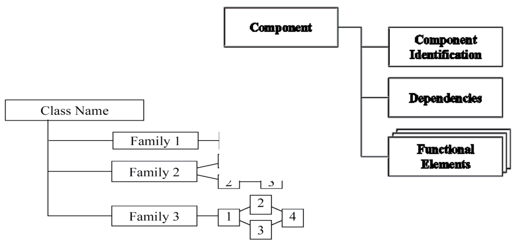
- The **functional requirements** are levied on those functions of the TOE that are specifically in support of IT security, and define the desired security behavior.
- Part 2 defines the CC functional requirements. Examples of functional requirements include requirements for identification, authentication, security audit and non-repudiation of origin.

Security Functional Components

- Class FAU: Security audit
- Class FCO: Communication
- Class FCS: Cryptographic support
- **Class FDP: User data protection**
- Class FIA: Identification and authentication
- Class FMT: Security management
- Class FPR: Privacy
- Class FPT: Protection of the TSF
- Class FRU: Resource utilisation
- Class FTA: TOE access
- Class FTP: Trusted path/channels

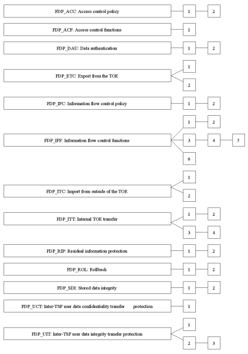
Security Functional Components

- Content and presentation of the functional requirements



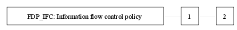
Decomposition of FDP

- FDP : User Data Protection



FDP – Information Flow Control

FDP_IFC.1 Subset information flow control



Hierarchical to: No other components.

Dependencies: FDP_IFF.1 Simple security attributes

FDP_IFC.1.1 The TSF shall enforce the [assignment: *information flow control SFP*] on [assignment: *list of subjects, information, and operations that cause controlled information to flow to and from controlled subjects covered by the SFP*].

FDP_IFC.2 Complete information flow control

Hierarchical to: FDP_IFC.1 Subset information flow control

Dependencies: FDP_IFF.1 Simple security attributes

FDP_IFC.2.1 The TSF shall enforce the [assignment: *information flow control SFP*] on [assignment: *list of subjects* and ***information***] **and all** operations that cause **that** information to flow to and from subjects covered by the **SFP.**

FDP_IFC.2.2 The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

Assurance Requirements

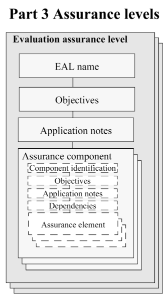
Assurance Approach

"The CC philosophy is to provide assurance based upon an evaluation (active investigation) of the IT product that is to be trusted. Evaluation has been the traditional means of providing assurance and is the basis for prior evaluation criteria documents. "

CC, Part 3, p.15

Assurance Requirements

- The **assurance requirements** are levied on actions of the developer, on evidence produced and on the actions of the evaluator.
- Examples of assurance requirements include constraints on the rigor of the **development process** and requirements to search for and analyze the impact of potential security vulnerabilities.
- The **degree of assurance** can be varied for a given set of functional requirements; therefore it is typically expressed in terms of increasing levels of rigor built with assurance components.
- Part 3 defines the CC assurance requirements and a scale of **evaluation assurance levels** (EALs) constructed using these components.



Assurance Components

- Class APE: Protection Profile evaluation
- Class ASE: Security Target evaluation
- Class ADV: Development
- Class AGD: Guidance documents
- Class ALC: Life-cycle support
- Class ATE: Tests
- Class AVA: Vulnerability assessment
- Class ACO: Composition

Assurance Components: Example

ADV_FSP.1 Basic functional specification

EAL-1: ... The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

EAL-2: ... The functional specification shall completely represent the TSF.

EAL-3: + ... The functional specification shall summarize the SFR-supporting and SFR-non-interfering actions associated with each TSFI.

EAL-4: + ... The functional specification shall describe all direct error messages that may result from an invocation of each TSFI.

EAL-5: ... The functional specification shall describe the TSFI using a semi-formal style.

EAL-6: ... The developer shall provide a formal presentation of the functional specification of the TSF. 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.

Degree of Assurance

(TSFI : Interface of the TOE Security Functionality (TSF), SFR : Security Functional Requirement)

Evaluation Assurance Level

- EALs define levels of assurance (no guarantees)

1. functionally tested
2. structurally tested
3. methodically tested and checked
4. methodically designed, tested, and reviewed
5. semiformally designed and tested
6. semiformally verified design and tested
7. formally verified design and tested

Assurance class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Development	ADV_ARC		**1**	1	1	1	1	1
	ADV_FSP	**1**	**2**	**3**	**4**	**5**	5	**6**
	ADV_IMP				**1**	1	**2**	2
	ADV_INT					**2**	**3**	3
	ADV_SPM						**1**	1
	ADV_TDS		**1**	**2**	**3**	**4**	**5**	**6**
Guidance documents	AGD_OPE	**1**	1	1	1	1	1	1
	AGD_PRE	**1**	1	1	1	1	1	1
Life-cycle support	ALC_CMC	**1**	**2**	**3**	**4**	4	**5**	5
	ALC_CMS	**1**	**2**	**3**	**4**	**5**	5	5
	ALC_DEL		**1**	1	1	1	1	1
	ALC_DVS			**1**	1	1	**2**	2
	ALC_FLR							
	ALC_LCD			**1**	1	1	1	**2**
	ALC_TAT				**1**	**2**	**3**	3
Security Target evaluation	ASE_CCL	**1**	1	1	1	1	1	1
	ASE_ECD	**1**	1	1	1	1	1	1
	ASE_INT	**1**	1	1	1	1	1	1
	ASE_OBJ	**1**	**2**	2	2	2	2	2
	ASE_REQ	**1**	**2**	2	2	2	2	2
	ASE_SPD		**1**	1	1	1	1	1
	ASE_TSS	**1**	1	1	1	1	1	1
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT			**1**	1	**3**	3	**4**
	ATE_FUN		**1**	1	1	1	**2**	2
	ATE_IND	**1**	**2**	2	2	2	2	**3**
Vulnerability assessment	AVA_VAN	**1**	**2**	2	**3**	**4**	**5**	5

Assurance Requirements

- EAL5 – EAL7 require **formal methods**.
- according to CC Glossary:

Formal: Expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts.

Security Functions

- The **statement of TOE security functions** shall cover the IT security functions and shall specify how these functions satisfy the TOE security functional requirements. This statement shall include a bi-directional mapping between functions and requirements that clearly shows which functions satisfy which requirements and that all requirements are met.
- Starting point for **design process**.

Summary

- Norms and standards enforce the application of the state-of-the-art when developing software which is
  - safety-critical or security-critical.
- Wanton disregard of these norms may lead to personal liability.
- Norms typically place a lot of emphasis on process.
- Key question are traceability of decisions and design, and verification and validation.
- Different application fields have different norms:
  - IEC 61508 and its specialisations, DO-178B.