EUROPEAN UNION AGENCY
FOR CYBERSECURITY

EUCC SCHEME STATE-OF-THE-ART DOCUMENT

Application of CC to integrated circuits

Version 1.1, October 2023

DOCUMENT HISTORY

Date	Version	Modification	Author's comments
October 2022	1.0	Creation	Version submitted to the ECCG Based on JIL-Application-of-CC-to-Integrated-Circuits-v3-0.pdf
October 2023	1.1	Addition of a document history section with a link to the SOG-IS document the state-of-the-art document is based on, where applicable	Based on ECCG comments received Approved for publication on 20/10/2023 by the ECCG and the European Commission

CONTENTS

1 INTRODUCTION

Today's increasing use of information technology has led to new methods of storage and processing of information. As a result of the packaging density on silicon, in the form of a microchip and improvements in performance, it is possible to process more and more information in parallel and at speed.

Complex microchips, which are able to process information, unfortunately introduce risks and dangers as well as huge advantages. Dependence on trouble-free functioning, as well as on the effectiveness of the protection measures, which have been carried out at the system and chip levels, has grown a great deal.

One must therefore be aware of the increased opportunities to test important information systems, including hardware against accepted criteria, in order to make the assurance of the security measures more transparent to the manufacturer, operator and user.

This state-of-the-art document supporting the EUCC scheme shall serve as a handbook for the application of CC to hardware components in respect of integrated circuits. This document will be of particular interest to manufacturers, evaluators and certifiers.

1.1 OBJECTIVE

The security properties of both hardware and software products can be certified in accordance with the CC. To have a common understanding and to ensure that CC is used for hardware integrated circuits in a manner consistent with today's state of the art hardware evaluations, the following chapters provide guidance on the individual aspects of the CC assurance work packages in addition to the Common Evaluation Methodology [CEM].

This document is applicable to the hardware of single ICs and covers assurance levels as defined in CC, with a focus on Security IC evaluation at the AVA_VAN.5 level.

2 ASSURANCE REQUIREMENTS FOR INTEGRATED CIRCUITS

2.1 INTRODUCTION

In applying CC to hardware components, two types of Target of Evaluation (TOE) may be considered:

- a TOE produced from a series of discrete parts on a printed circuit board or as a hybrid through different or several dice on one carrier;
- a TOE produced as an individual integrated circuit (IC).

The following CC assurance aspects are applicable to the hardware of single ICs.

In general, logical functionality in an IC can be implemented in simple PLD structures (Programmable Logic Device), FPGA structures (Field Programmable Gate Array), as an ASIC (Application Specific Integrated Circuit), or as well as a customer IC.

In respect of security applications, intelligent memory ICs with a hard-wired security logic (e.g. public transport cards, building access control or micro-controller based ICs in an ASIC design (e.g. electronic purse ICs) are used mainly for the elementary core components of security functionality.

Contemporary memory in ICs is based on EPROM, E2PROM or flash cells. This enables the non-volatile storage of data. Security logic can be utilised to implement identification and authentication, access control and internal IC sequence controls.

Micro-controller-based ICs offer the possibility of carrying out independently complex processes controlled by an IC operating system. Items, which also belong to the aforementioned functionality, comprise accountability functions and services such as encryption, random number generation and digital signature, functionality that is implemented in hardware as well as software.

The mechanisms for the protection of software and operational data in various memories and the internal sequences are realised through the hardware of an IC (e.g. by means of certain technical measures and technological features) in order to support logical functionality.

An operating system of a micro-controller IC is contained (placed) in a ROM and/or an E2PROM memory, and is protected from disclosure and modification during the operational phase by the technical or technological properties of the hardware. While technological properties are inherent in a TOE, technical properties depend on the design of the TOE.

2.2 CC APPROACH FOR EVALUATION

The CC prescribes a variety of assurance activities, such as TOE specific as design analysis, guidance documentation, vulnerability analysis, penetration testing, on the one hand and the examination of development and production environment on the other hand. Whilst there are differences in terminology used, fundamentally the two sets of criteria are very similar in terms of specifying assurance requirements, and in their underlying philosophy.

A unique feature of the CC is the introduction of the Protection Profile concept. A Protection Profile can be characterized as a generic Security Target for a particular class of TOE (e.g. Security ICs, as in the case of [BSI-CC-PP-0084-2014][1], which defines a recognized standard to which conformance may be claimed. Whilst it is not mandatory to claim conformance to a Protection Profile, the availability of such standards provides the potential to reuse evaluated material in a Security Target, thereby considerably easing the process of producing this particular evaluation deliverable for the developer. For the user, conformance to one PP by several STs (i.e. by different products) allows comparison of these products on an equal basis. Especially in the case where there is a single, well-established Protection Profile for that domain, this is a strong mechanism that gives the user more choice and better cost-efficiency.

A Protection Profile (PP) may be defined, evaluated and certified in advance of a real TOE Evaluation and can be referenced within the Security Target of the real TOE Evaluation. There are two types of conformance: demonstrable and strict. In the demonstrable case, the security functional requirements (SFRs) of the Security Target are argued to be similar to the ones in the PP. In the case of strict conformance, the ST is conformant to a PP only if it is compliant with all parts of the PP. The PP referenced sets the level of conformance required. In the Security IC domain, strict conformance is most common.

The following chapters provide guidance for hardware TOEs that have to be evaluated under Common Criteria (CC).

The evaluation of the IC comprises the following activities:

- Security Target,
- Development,
- Tests,
- Guidance including operation
- Life-cycle support, including configuration management and delivery
- Vulnerability assessment.

The above mentioned activities correspond to certain assurance classes defined in the CC. The following table shows the Assurance Class / Assurance family breakdown and mapping.

Table 1 - Assurance family breakdown and mapping

Assurance Class	Assurance Family	Abbreviated Name
Class APE: Protection Profile Evaluation		APE[2]
Class ASE: Security Target Evaluation		ASE[3]
Class ADV: Development	Security Architecture Functional specification TOE Design Implementation representation TSF internals Security policy modelling	ADV_ARC ADV_FSP ADV_TDS ADV_IMP ADV_INT ADV_SPM
Class ATE: Tests	Coverage Depth Functional tests Independent testing	ATE_COV ATE_DPT ATE_FUN ATE_IND
Class AGD: Guidance documents	Operational user guidance Preparative procedures	AGD_OPE AGD_PRE

[1] Security IC Platform Protection Profile with Augmentation Packages.
[2] The assurance class APE is split into several families (see CC).
[3] The assurance class ASE is split into several families (see CC).

Class ALC: Life cycle Support	CM capabilities CM scope Delivery Development security Flaw remediation Life-cycle definition Tools and techniques	ALC_CMC ALC_CMS ALC_DEL ALC_DVS ALC_FLR ALC_LCD ALC_TAT
Class AVA: Vulnerability assessment	Vulnerability analysis	AVA_VAN

The following elements give guidance on how to use the assurance components of CC assurance classes for hardware IC TOEs (e.g. a Security IC Platform).

Where applicable, all dependency requirements as outlined within CC of selected assurance components will have to be fulfilled.

2.3 CC PROTECTION PROFILE (CLASS APE)

Unlike a ST, which describes implementation oriented security; a PP describes abstract security requirements. For instance, the requirement for a random number generator (RNG) with an un-specified quality could be expressed in a PP, and a compliant ST could then state to what quality level the particular TOE provides random numbers.

The majority of guidance applicable to a ST applies equally well to a PP (see the next chapter) for example, definition of scope and boundary of TOE, environmental assumptions, threats, security objectives.

The reference PP related to IC and conformant to CC is [BSI-CC-PP-0084-2014][4]. It has been developed by a community of semi-conductor manufacturers and takes advantage of wide experience in Smartcard security.

2.4 CC SECURITY TARGET (CLASS ASE)

2.4.1 Objectives

The Security Target (ST) for a TOE is the basis for the evaluation and shall be agreed between developer and evaluator. The audience for the ST is not confined to those responsible for the production of the TOE and its evaluation, but may also include those responsible for managing, marketing, purchasing, installing, configuring, operating and using the TOE.

The CC describe in details the content and presentation requirements of the ST.

A Security Target comprises the following:

- a precise description of the security problem solved by the TOE and its environment in terms of threats, any assumptions, organisational security policies and intended use;
- a description of the security objectives for the TOE and for the environment in order to determine whether the security objectives counter the identified threats, achieve the identified organisational security policies and adhere to the stated assumptions.
- Protection Profile claims if any exists;
- a description of the security functional requirements and security assurance requirements of the TOE;
- a summary specification of how the TOE implements the security functional requirements;
- a rationale giving justification for transformation of the security problem to the security objectives to the security requirements.

Following sections provide observations concerning individual requirements.

2.4.2 Input

The developer shall provide the document ”Security Target”.

2.4.3 Requirements

TOE description

[4] For more PPs refer to the ENISA website dedicated to cybersecurity certification..

The Security Target shall include a precise description of the Target of Evaluation (TOE) in terms of hardware, software and firmware components. Reference to technological parameters is also important. The TOE description shall explicitly state the nature of any dedicated test software (either embedded software or software outside the integrated circuit).

The general security characteristics of the hardware shall also be described. A reference to the hardware datasheet would be appreciable. All possible configurations or intended use of the chip shall be identified.

The TOE needs to be clearly identified and separated from its technical and operational environment. The ST shall uniquely reference the TOE.

Since the hardware parts of an IC are both physically and functionally difficult to separate from one another without additional information, it is not possible to exclude parts of the hardware from the TOE; it is therefore sensible to define the whole of the IC hardware as the TOE. In the case of certain parts of the IC being outside the TOE, a clear, logical and physical interface must exist. The inclusion of strongly hardware-oriented software/firmware in the TOE is appropriate. It would be sensible to look at an IC in its entirety and not only at the hardware or only at the software.

Security Environment
The security environment of the TOE comprises the operational environment after delivery as well as the technical environment in the different phases of the lifecycle of the TOE.

Therefore, a precise description of the TOE lifecycle is required or should be referenced. The boundaries of the TOE in terms of lifecycle shall be defined.

The Security Target shall explicitly state which phases of the life cycle are under the scope of the evaluation and which phases are excluded; the phases where the TOE is being developed and manufactured shall always be within evaluation scope.

In contrast with purely software TOEs, as in the cases noted here, this determination is only possible with precise knowledge of the manufacturing process. However, this statement also has a direct influence on the threat and attack scenarios in the operation of the TOE, which could be adopted in the context of the evaluation of the TOE. In the case of an IC TOE, the cut-off for evaluation could be after testing the IC as a die (at the earliest), or upon completion of packaging and associated testing. Optionally, the Security Target may include further phases of the IC such as micromodules assembly, pre-personalisation and personalisation phases.

The CC explicitly require that threats be characterized in terms of an identified threat agent, the asset at risk, and the attack. Thus it is necessary to define and enumerate all subjects in terms of roles and the assets for which specific protection either by the TOE or its environment is required and with reference to the lifecycle phases of the TOE.

Any assumptions placed on the environment, with which the TOE or its environment must comply, have to be identified.

Assumptions relating to the operation of the software, which is not part of the TOE are essential. These may include:

- integrity protection software, e.g. for responding to sensors, or to watchdog timer interrupts;
- implementation of algorithms that are DPA-resistant;
- fault-handling software (e.g. protecting against inducing faults to enable a differential fault analysis attack on cryptographic keys).

Regarding the operational phase of the TOE, any assumption on the security aspects of the environment in which the TOE will be used or is intended to be used shall be described. Generally for a Security IC, no specific assumption for the TOE and its environment during the end-user phase (operational environment) has to be defined since this environment is a public one. However, if the TOE comprises only IC-hardware, there may be important security assumptions for the usage-phase of the TOE.

With reference to the lifecycle phases which are beyond the scope of the evaluation, there should be information about who is able to use the TOE after delivery and in what operational modes it is possible to use it. In this context, the actions of all personnel who come into contact with the TOE after delivery need to be examined. Therefore, specific assumptions about the behaviour of such personnel need to be defined and an identification of operational roles involved is necessary. Note: hardware may add a variety of roles that are IC-specific, e.g. there may be different approaches to personalization and enablement; such hardware-specific roles may need to be documented outside of the ST.

As an example for specific assets for a Security IC this may include:

- IC specific data including personalisation data and cryptographic keys,
- smartcard embedded software,
- IC dedicated software,
- Specific Application data like keys, authentication data or access control information.

A distinction might be useful between primary assets - such as the data stored and operated by the Security IC Embedded Software, the Security IC Embedded Software itself when stored and in operation, and the security services provided by the TOE for the Security IC Embedded Software - and secondary assets which, if compromised, could be exploited to compromise a primary asset. Secondary assets do not have any intrinsic value as such, but instead derive value from the primary assets. This distinction would allow a separation of high and low-level assets, which in turn will help to structure the statement of threats and thereby lead to a better understanding of the security objectives and security requirements to be met by the IC.

However, the CC do not mandate that low-level or secondary assets are identified in order to drive the selection of security objectives and requirements. For example, integrity protection SFRs can be included simply to help achieve a security objective for protection of a high-level or primary asset - with the security requirements rationale explaining the purpose of such SFRs.

Assumed threats to the assets shall be described. The CC require that threats defined in the ST, are not directed at the identified security objectives, but rather are addressed by the security objectives. It should also be noted that the CC explicitly require that threats be characterized in terms of an identified threat agent, the asset at risk, and the attack (describing such aspects as attack methods employed, vulnerabilities exploited, and opportunity).

It shall be a description in terms of damages to the assets rather than attack paths, which could not be completed in the ST. The threats could be described in terms of:

- unauthorized disclosure of assets;
- unauthorized use of assets;
- unauthorized modification of assets.

To be able to understand the threats defined for the environment of the TOE in certain phases of the lifecycle, the TOE's development and production environment shall be described.

Beside logical functionalities, technical and technological properties can be attacked during the operational phase of the TOE, too. The corresponding threats to the defined assets can therefore be formulated (e.g. selection of objects also by means of physical attacks, operation of the TOE outside specific parameters, such as voltage, frequency and temperature).

With respect to determining specific threats, it should be noted that attacks on ICs during production processes, and in particular during test phases, are possible and shall be considered for the relevant life-cycle phases within the relevant assurance class ALC. They may result in vulnerabilities in the development of the TOE identified by the evaluator during the conduct of evaluation activities for class ALC.

Specifications of organizational security policies depend essentially on the applications in which the TOE is incorporated. Generally speaking, for a pure hardware Security IC evaluation, no specific organizational security policy has to be defined.

Security Objectives
The CC require that security objectives be specified within the ST for both the TOE and the environment that are necessary to counter the threats and uphold the identified assumptions and organizational security policies.

The objectives shall be clearly stated to permit a clear mapping back to the relevant threats. They could be derived from the following:

- resistance against physical manipulation;
- resistance against physical probing;
- protection from information leakage (naturally occurring and attacker-induced);
- protection of test functionality;
- storage of data by test personnel;
- providing random numbers.

Technical and technological properties of the IC beyond the objectives, i.e. those providing self protection, will be addressed in the ADV_ARC evaluation activity. The ST author can choose to highlight these properties to users in the TOE Summary Specification (as part of ASE_TSS.2).

Additionally, there may be a need for specific security objectives on the environment to ensure that assumptions concerning dependencies on software are upheld.

Security objectives for the environment within certain lifecycle phases might be satisfied by measures for the environment of the TOE evaluated by the assurance requirements for the development process of the TOE. (e.g. development security).

Security requirements
Security requirements shall be defined within the ST using the functional and assurance components specified in the CC. In some cases, if predefined functional components of the CC are not applicable, new IC-specific components might be defined within a ST.

It is required that the security functional requirements (SFR) and assurance requirements (SAR) on the TOE are needed to meet the identified security objectives for the TOE.

SAR:
The required level of attack potential for the vulnerability analysis is expressed in CC requirements by selection of a certain AVA_VAN assurance component. This component defines the baseline for the protection of the TOE in terms of attack potential against which the vulnerability analysis of the TOE will be judged.

The evaluators' independent vulnerability analysis builds on information gained from all other evaluation activities and goes beyond the security architecture description (often seen as "the developer's vulnerability analysis"). The main intent of the evaluator analysis is to determine that the TOE is resistant to penetration attacks performed by an attacker possessing a basic (for AVA_VAN.1 and AVA_VAN.2), enhanced-basic (for AVA_VAN.3), moderate (for AVA_VAN.4) or high (for AVA_VAN.5) attack potential.

By way of example, to ensure resistance against high attack potential for vulnerability analysis, the AVA-class to be selected is AVA_VAN.5. Additionally, the components which AVA_VAN.5 depends on have to be required within the SAR. For more information on attack potential refer to the state-of-the-art document supporting the EUCC scheme APPLICATION OF ATTACK POTENTIAL TO SMARTCARDS AND SIMILAR DEVICES.

SFR:
Some guidance on the use of such components in a PP or ST for an IC might be helpful, although there are now a number of PPs in existence for ICs which can be usefully referenced (indeed, a need to comply with such PPs may pre-determine which functional components must be used in the ST).

Possible CC Security Functional Requirements (SFRs) for an IC comprise for example components from the following classes: FIA, FMT, FCS, FDP, and FPT.

It is important that the selected functional components be tailored to the extent necessary to demonstrate that the TOE security objectives are met. This applies to both PPs and STs (but especially the former, where operations on functional components can be left uncompleted, thus resulting in SFRs, which are too generic).

A further issue is the CC requirement that SFRs are actually testable. For guidelines on testing refer to ATE below.

The FPT_PHP components are used to express requirements for protection of the TOE from physical tampering attacks and require the TOE to implement functions to respond to these attacks - whether by:

- providing the capability to detect an attack (FPT_PHP.1); or
- detecting and providing notification of an attack (FPT_PHP.2); or
- responding automatically to resist an attack (FPT_PHP.3).

The selection of these components might be adapted to the situation in place. For example, responding automatically to resist an attack (FPT_PHP.3) may be refined as assuming that there might be an attack at any time and therefore providing countermeasures at any time because the TOE´s technical properties may not be able to detect the attack but are activated in place permanently. For example, permanent protection against Differential Power Analysis is required, ensuring that the SFR could not be violated or bypassed at any time.

In the evaluation of a composite TOE (IC hardware plus software: operating system, application software and IC dedicated software) it may be applicable to select functional components for an information flow control policy through the use of functional components from the FDP_IFC and FDP_IFF families. These components apply to certain parts of the software which are part of the TOE (for example such requirements are placed on the OS and hence on the integrated platform comprising IC and OS; in [BSI-CC-PP-0084-2014] such components are applied to IC dedicated software).

A ST or PP may modify selected CC Part 2 components to be more meaningful to smartcards, (however, it should be noted that modified requirements like this need to be proven in practice) such as:

- a deviation from audit data generation component, FAU_GEN.1 to exclude the requirement for date and time in the audit record. However, this requirement is only achievable if an externally trusted time source exists and trust can be preserved in the record;
- a refinement of FPT_TST.1, self-processing, to include card blocking functions.

Operations on requirements:
The Security Target shall explicitly perform all the operations (assignment, iteration, selection, and refinement) of the security requirements. These operations may concern both functional security requirements as well as assurance security requirements. As a minimum, all the operations of assignment and selection of functional security requirements shall be performed.

For each selection, the Security Target author shall select the appropriate item in the selection list. For each assignment, the Security Target author shall specify the appropriate item. Any guidance could be found in the annexes of the CC.

For ICs, it is important that security aspects of design and implementation, which are not necessarily functional in nature, be addressed by the evaluation.

TOE summary specification (TSS)
The TOE summary specification provides a high-level description how the TOE implements the functional requirements (SFR) as defined in the ST. The ST author can choose to highlight technical and technological properties of the IC to users in the TOE Summary Specification (as part of ASE_TSS.2).

PP claims
The Security Target shall explicitly claim compliance with any Protection Profile if applicable. In this case the ST must claim the conformity explicitly.

[BSI-CC-PP-0084-2014] which has been developed by a community of semi-conductor manufacturers could be referenced[5] in the Security Target.

It shall be noted that claims of partial compliance to a PP is not permissible under the CC. PP compliance can be demonstrable or strict, as defined in the Protection Profile referenced. In the Security IC domain, strict PP compliance is preferred.

In case of any PP compliance, the Security Target does not need to repeat statements of security requirements included in the PP that are unmodified for the Security Target. Nevertheless, it could be easier to have an independent document.

If, however, the PP includes uncompleted operations, which is the case for [BSI-CC-PP-0084-2014], completing these operations is the responsibility of the Security Target author.

Rationale
The security target is fundamental to set up an effectiveness view since it lists the intended use of the IC, the operational environment, the assumed threats, objectives, functional and assurance requirements and the TOE summary specification as discussed above.

The CC require a rationale, which demonstrates that a TOE conformant with the ST will effectively address all relevant aspects of the ‘security problem’ defined by the Statement of Security problem definition. The ST rationale presents the analysis in a step-wise manner:

- firstly, the security objectives for the TOE and its environment must be shown to be suitable to counter the identified threats (transposed by certain attack scenarios) and uphold all identified policy needs and assumptions. If applicable, scenarios of physical attacks on the hardware can be of significance. Assumptions made relating to the software operation are essential;
- secondly, the security requirements must be shown to be suitable to satisfy the TOE security objectives, and be mutually supportive and provide an integrated and effective whole (binding). Therefore, the analysis should consider combinations of SFR where some of them may be logical and others technological and technical requirements. The analysis should also consider relations between assurance requirements and

[5] For a list of applicable PPs refer to the website on European cybersecurity certification schemes maintained by ENISA.

objectives for the operational environment of certain phases of the lifecycle like composite product integration and personalization phases.

Binding of hardware- and firmware-functionalities is to be taken into consideration, depending on the scope of the TOE. For an IC, the Security Target should describe the assumptions made relating to the software operation.

Detailed aspects addressing the issue of indirect attacks against the IC (e.g. bypassing or contradicting the TSF) should be part of the vulnerability assessment (see class AVA). The developer supports this with his view on how indirect attacks in the form of bypass or tampering are countered (in the security architecture description, evaluated in ADV_ARC).

In the evaluation of a composite TOE, the analysis should discuss the interrelationships between the software and hardware parts of the TOE to demonstrate that they are mutually supportive in helping to meet the Security Target for the composite TOE. This will not only involve discussion of dependencies of the IC on the software such as those listed above, but also the software dependencies on the hardware, including tamper-resistance aspects.

Analysis is mostly done by providing mappings in combination with informal arguments of the mapped items and explanations on relations between certain items.

2.5 DEVELOPMENT (CLASS ADV)

The assurance class ADV defines requirements for the stepwise refinement of the TOE Security Functionality (TSF) from the TOE security functional requirements (SFR) in the ST down to the actual implementation, and defines requirements for the description of architectural oriented features and internal structure of the TOE (ADV_ARC, ADV_INT). Each of the resulting TSF representations provides information to help the evaluator determine whether the functional requirements of the TOE have been met.

The technical description of the TOE is always accompanied by mapping the higher-level to the lower-level of the TOE representations.

2.5.1 Architecture (ADV_ARC)

2.5.1.1 Objectives

The CC introduce the security assurance family Security Architecture (ADV_ARC). Its objective is described as follows:

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

The family ADV_ARC requires the TOE security architecture to describe the self-protection, domain separation and non-bypassability principles. The Security Architecture shall also describe the secure TOE security functionality (TSF) initialisation.

2.5.1.2 Input

The developer shall provide a security architecture description of the TSF at a level of detail commensurate with the description of the SFR-enforcing abstractions described in the TOE design document.

The security architecture description shall demonstrate the effectiveness of self-protection and non-bypassability.

2.5.1.3 Requirements

Non-bypassability is a property that the security functionality as specified by the SFRs is always invoked and cannot be circumvented when appropriate for that specific mechanism. The CC discuss the bypassability of the SFR-enforcement through different interfaces and the information provided in the functional specification (ADV_FSP) and TOE design (ADV_TDS) about the operations and information available trough these interfaces. This includes all logical and physical programming and communication interfaces of the IC as well as the surface of the IC.

Self-protection refers to the ability of the TSF to protect itself from manipulation from external entities that may result in changes to the TSF, so that it no longer fulfil the SFRs. Self-protection of the TSF will be achieved by:

- self-protection of TSF mechanisms: the ability of a TSF mechanism to protect themselves against direct attacks to interfere, to manipulate or to disable this mechanism;

- binding of TSF mechanisms: the ability of the TSF mechanisms to work together in a way that is mutually supportive and provides an integrated and effective whole.

Example of self-protection of a TSF mechanism is the protection against physical probing of a security IC to manipulate or to disable TSF features. In some cases the physical protection mechanism must resist such attacks independent on any other TSF mechanism. In other cases the physical protection mechanism detects such attacks and the operating system reacts on it entering a secure state.

An example of binding of TSF mechanisms in case of smart cards is the combination of hardware and software TSF mechanisms:

- to ensure the stable correct execution of the embedded software under specified operational conditions;
- to detect errors of the execution caused by perturbation;
- to enter a secure state if detected errors can not be automatically corrected.

The security architecture description shall describe how the TSF initialization process is secure (cf. ADV_ARC.1.3C). The information provided in the security architecture description relating to TSF initialisation is directed at the TOE components that are involved in bringing the TSF from the “down” state (e.g. power-off) into an initial secure state (i.e. when all parts of the TSF are operational). The TSF may have parts providing their security function when the TOE is not running. E.g. the physical protection of a security integrated circuit shall resist tampering attacks according to FPT_PHP.3 even if power is off. Other parts of the TSF may be activated during start-up of the TOE before or at the same time when the relevant TOE functionality is activated. E.g. sensors shall control the environmental conditions for the normal secure operation of a security IC at time the operating system of the smart card starts-up. The operating system shall check the integrity of stored TSF data before relying on it. For security IC the secure initialisation process of the TSF covers power-on start-up, entering and wake-up from power save modes or any kind of reset.

Domain separation is a property whereby the TSF creates separate security domains on its own and for each untrusted active entity to operate on its resources, and then keeps those domains separated from one another so that no entity can run in the domain of any other (cf. CC). Security domains refer to environments supplied by the TSF for use by potentially-harmful entities (cf. CEM). The environment provided by the TSF of the IC to an entity at the programming interface may comprise resources like

- input and output ports / interfaces to interact with external entities (users) or processes (subjects) of the IC dedicated or embedded software;
- address space to access operational memory and functional registers;
- commands of CPU or cryptographic coprocessors available for execution by an entity;
- services provided by the TSF like random number generation.

The TSF may use specific resources for their own security domain only like e.g. sensors for environmental failure protection being outside control of the embedded software. The TSF may share resources with other entities like e.g. random number generator used by the TSF for randomization of cryptographic coprocessor computation and used by the embedded software for key generation. The TSF may control access to resources by entities of different security domains but not used by its own like privileged CPU commands. The TSF may provide security capabilities for the embedded software to enforce their domain separation e.g. a memory management unit.

2.5.2 Functional specification (ADV_FSP)

2.5.2.1 Objectives

The functional specification describes the TSF interfaces (TSFI), and must be a complete and accurate instantiation of the TOE security functional requirements as defined in the ST.

The goal of the ADV_FSP family is the description and analysis of the external interface to the TOE. Users of the TOE are expected to interact with the TSF through this interface. The TSFI consist of all means for users to invoke a service from the TSF (by supplying data, signals, energy or physical effects that are processed by the TSF) and the corresponding responses to those service invocations. These service invocations and responses are the means of crossing the TSF boundary (cf. CC). All interfaces crossing the border of the TSF including interfaces to the non-TSF subsystems of the TOE are considered as TSFI. The TSFI description provides necessary information to conduct testing.

The components ADV_FSP.2 and higher describes in increasing levels of details all TSFI. At lower-level components, the developers may focus their documentation (and evaluators focus their analysis) on the more security-relevant aspects of the TOE by characterization of the TSFI as SFR-enforcing, SFR-relevant and SFR-non-interfering. The components ADV_FSP.4 and higher typically used for security IC describe all interfaces on the same level of details and allow for deeper analysis that the interfaces do not provide functions in a way contradicting the SFR defined in the ST.

2.5.2.2 Input

The developer shall provide the functional specification.

The external interfaces are usually described in the data sheet for the IC. The ISO/IEC 7816 standard is of relevance in most cases. In addition, the die description shall be given.

2.5.2.3 Requirements

The functional specification usually uses developers' terminology. The level of detail required for the specification has to be correlated to the coverage of functional tests (ATE_COV) and to the external interface description within the guidance documents (AGD_OPE / AGD_PRE).

Specification of functional details of the TOE security functions has to be provided within the Functional Specification. More detailed representation levels map these functional details to the defined subsystems or modules.

External interfaces of a security IC can be classified as[6]:

- Program interfaces – the logical interface to the software/firmware which is stored on the IC and is not part of the TOE, but which runs on the IC hardware under consideration (e.g. the triggering of an interrupt via hardware, test software interfaces).
- Communication interface – the logical interfaces (e.g. instruction set, special function register specification, memory map) and the physical interfaces of the IC (IC contacts with serial I/O and supply or contactless interface or both), which guarantee the connection to the outside world within the operational environment and the operating system / application programming environment.
- IC surface – an explicitly defined continuous perimeter that establishes the physical bounds of the TOE and contains all the hardware, software, and/or firmware components of the TOE. The IC surface shall be described and examined for physical protection (cf. SFR of the FPT_PHP family) and completeness of the logical and physical interface description, including areas of emanation and for irradiation.

For a security IC, the physical entry or exit point of the TOE (ports) shall be described that provides access to the TOE for physical signals, represented by logical interfaces, including power supply. A port may provide more information than necessary for the TSFI. This additional information may bypass the security functionality intentionally provided through this interface (e.g. by a side channel). Simply observing external interfaces from the point of view of their logical behaviour is unlikely to be sufficient. Externally adjustable operational parameters and their limits should also be investigated because direct attacks or vulnerabilities may result. The functional specification shall trace the SFRs to TSFIs.

The IC Dedicated Test Software may be delivered as part of the TOE to support testing of the TOE during production and may not be usable after TOE Delivery. In this case the IC Dedicated Test Software (or parts of it) is seen only as a "test tool", which does not provide security functionality for the operational phase of the TOE. Their use shall be described in the guidance documentation for the tester but not necessarily in the Functional Specification. However, it must be verified that it cannot be abused after TOE Delivery: this is evaluated according to the CC assurance family AVA_VAN.

The Functional Specification shall specify operating conditions of the security IC. These conditions include but are not limited to the frequency of the clock, the power supply, and the temperature. The security IC should respond to violation of operating condition threatening the correct execution of the embedded software by entering a secure state.

Semi-formal notation

Where the functional specification is required to be semiformal, it can take the form of

- tables, where every column is assigned to a specific bit of a specific register and informal text explains their effect;
- block diagrams, where all block diagrams used abbreviations and arrows to define the direction of the data flow;
- mathematic formula, e.g. Boolean logic expression describing the combination of signals or an inequation for the specification of the control of various thresholds;
- pseudocode used similar to the programming language and which does not include any unclear structures;

[6] Note, any interface of the TSF to a non-TSF portion of the TOE is not viewed as TSFI.

- assembler code describing program sequences to specify the behaviour and intended usage of specific components.

The informal documentation of the technical and technological properties, as well as their integration into the structure and the realization of SFR are necessary.

2.5.3 TOE design (ADV_TDS)

2.5.3.1 Objectives

The objective of these requirements is to provide both context for a description of the TSF, and a thorough description of the TSF. The design documentation shall provide sufficient information to determine the TSF boundary, and to describe how the TSF implements the SFR.

The design requirements are intended to provide information (commensurate with the given assurance level) so that a determination can be made that the security functional requirements are realised.

The TOE design provides a description of the TOE and TSF in terms of subsystems as major structural units with functional coherence, provides a description of the interaction of these structural units, and is a correct realisation of the functional specification.

The TOE design provides a description of the TSF in terms of modules as most specific description of functionality. The description of the modules shall provide sufficient details that the developer should be able to implement this part of the TOE described by the module with no further design decisions.

2.5.3.2 Input

TOE design information shall be made available.

2.5.3.3 Requirements

TOE vs. TSF
The TOE design describes the structure of the TOE[7] in terms of subsystems and identifies the subsystems of the TSF. The TSF includes all parts of the TOE that contribute to the satisfaction of an SFR in the ST (in whole or in part) and the security architectural principles of TSF self-protection, domain isolation, non-bypassability and secure initialization (see ADV_ARC). Any part of the TOE not being part of the TSF must not prevent the TSF from satisfying the SFR in the ST.

If TOE subsystems are separated TSF subsystems, the justification about the clearness and effectiveness of the separation should be based on logical and physical dependencies. A maximal independence of subsystems within an IC TOE could be possible if there were no or only minimal physical overlaps and logical dependencies between the individual subsystems and the interfaces are clearly defined.

Basic structure of the TSF in terms of subsystems and modules:
The developer's choice of subsystem definition and at level ADV_TDS.3 or higher the refinement into TSF modules within each subsystem are an important consideration in making the TOE design useful in understanding the TSF intended operation. The number of subsystems and modules within subsystems together with the description of their interaction, interfaces, and the purpose of the modules has to be appropriate and sufficient for the evaluator to gain the necessary level of understanding how the functionality of the TSF is provided.

If the IC design is managed through a classical process of hardware drawings, the development process depends essentially on the technologies used (specific method and tools) and can be described by refining design in terms of subsystems into a sufficient level of detail in terms of modules to implement the TOE. If the IC design is managed through a hardware description language (HDL), the decomposition is similar to those used in classical software development. The construction plans and functional descriptions, which result from the use of an HDL tool and a CAD tool, can be used directly during the construction of the TSF design, but needs only to be presented fully for ADV_TDS.4 or higher.

The TOE design provides a top level design specification in terms of subsystems of the TOE and the TSF. The complete data book comprising the complete description of the chip may be considered to support these requirements. A block diagram, which originates in the design and conception phase, as well as an informal description, can be an integral part of the TOE design description in terms of subsystems. Typically, the

[7] Here and in the following sections the term "TOE" is used for the software, firmware and/or hardware part of the TOE. The guidance documentation as part of the TOE is discussed in chapter 2.10 Guidance Documents (Class AGD).

documentation needed for the subsystems may be described as a mapping of the major architectural components to the physical devices performing specific functions (e.g. CPU, RAM, ROM, Bus and I/O elements) and the interaction among the subsystems.

In many cases, components which represent the general structure of an IC TOE can be definite logical units; they are possibly even implemented as a physical unit on the IC. Examples comprise: memory, data/address bus–memory interface, arithmetic block, contact interface, watchdog timer, sensors with analysis logic, controls for the voltage supply, logic blocks for access controls or authentication for memory ICs with security logic, a micro controller block on micro controller ICs.

The TOE design at level ADV_TDS.3 or higher requires a refinement of the TSF subsystems into modules. Modules are described in detail in terms of the function they provide (the purpose); the interfaces they present; the return values from such interfaces; the interfaces (presented by other modules) they use; and a description of how they provide their functionality (one possible way to describe the functionality is an algorithmic description) (cf. CC).

The form of description of the functionality and how the module provides them depends on the functionality. E.g. the TOE design may provide an algorithmic description for calculation of a cryptographic coprocessor as well as an informal description of the physical principle used for a sensor or a noise source of a physical random generator.

The design elements necessary for the construction of the TOE are:

- logical plans (which for example consist of analogue cells, standard cells, gates, transistors and diodes) or corresponding HDL-representations in order to realise individual functionality as well as security mechanisms;
- specification of the physical design which describes the requirements for the organisation of the physical components (e.g. module placement, layer order, routing specifications).

In the case of an IC, the actual logic and layout plans will have been derived from modules, whereby the separation of the modules has to fulfil the testing requirements. Interfaces between modules must be described especially carefully, since there are strong dependencies between them in an IC. Functionality, which runs in parallel, should be considered with the description of the interfaces. The timing of the module interfaces should be described if they are accessible from the outside (e.g. pads) for tests.

Since the security properties of an IC TOE can consist of logical functionality as well as technical and technological properties, it is necessary to document the general structure of the architectural components as well as to explain the technical and technological structure (general layout rules of the physical design: technology, number of layers, bus routing) because of their importance to the hardware security properties. A protective layer could, for example, be seen as a component of the general structure of the physical composition of the TOE.

The description of the TSF in terms of SFR-enforcing, SFR-supporting and SFR-Non-interfering subsystems and modules
The TOE design shall designate the TSF subsystems - and additionally at level ADV_TDS.3 and higher the modules - as SFR-enforcing, SFR-supporting and SFR-non-interfering. At lower-level components, developers may focus their documentation (and evaluators focus their analysis) on the more security-relevant aspects of the TOE, i.e. starting with SFR-enforcing and going further to SFR-supporting components up to SFR-non-interfering components. It should be noted that even when more or even complete information is required at higher level components, it is not required that all of this information be analyzed in the same level of detail. The focus should be in all cases on whether the necessary information has been provided and analyzed.

Mapping of the SFR to physical subsystems may not be easy to do (e.g. which subsystem really does process the security functions, the CPU or a CPU in conjunction with its associated memory and bus?). This is because within the IC itself there are strong dependencies between various physical components at the implementation level, which complicate an effective separation in the sense of the criteria. Consequently, it is mostly necessary and may be easier for some hardware TOEs to classify all of the subsystems of an IC TOE at the level of the high-level design as SFR-enforcing.

For example Test-ROM firmware could be classified as “SFR-non-interfering subsystem” because it does not contribute to the SFR (but is necessary for the manufacturer test) and is deactivated in the operational phase of the TOE. Another example, depending on the specific security functionality of a TOE, may be a standard peripheral unit, e.g. a timer, if separation can be shown.

Evidence of how the SFRs are provided

The evaluator shall determine that the design is an accurate and complete instantiation of all security functional requirements (cf. ADV_TDS.x.2E). Assigning the SFR to subsystems or even modules may be especially difficult, since individual components not only provide the realization of a single SFR and very strong interactions and dependencies between components may exist. Mapping of the SFR of the ST via the Functional Specification to physical subsystems and modules may not be easy to do as mentioned above. For this reason, a description of the functional flow of security functions implementing the SFR to defined subsystems is of particular significance.

The SFRs of the ST expressing security properties might be realized and traced to technological properties of the TOE as described in the design.

Semi-formal notation
Where the TOE design is required to be semiformal, it can take the form of block circuit diagrams or hardware description language documents (HDL – hardware description language). In many cases, however, a hardware description language would first be used at the Detailed Design level.

Meaningful graphical representations of the technical or technological properties as part of the security mechanisms of the TOE may be considered equivalent to semi-formal representation.

The informal documentation of the technical and technological properties, as well as their integration into the structure and the realization of security mechanisms are necessary.

2.5.4 Implementation Representation (ADV_IMP)

2.5.4.1 Objectives
The objective of these requirements is to determine whether the implementation representation is sufficient to satisfy the SFR of the ST and is a correct realisation of the TOE design in a form that can be analysed by the evaluator.

It is the least abstract representation of the TSF and captures the detailed internal workings of the TSF in terms of source code, hardware diagrams and/or hardware design language or layout data, etc., as applicable.

2.5.4.2 Input
The implementation representation shall be made available by the developer.

2.5.4.3 Requirements

Implementation representation for security IC
The TOE implementation representation corresponds to the following:

- hardware diagrams for analogic blocks;
- HDL statements for all synthesised components;
- if applicable, source code for all dedicated/embedded software;
- physical design information like layout and mask plans describing the implementation of the physical components.

The complete implementation information and layout shall be made available to the evaluator in a form used by the development personnel. This will imply in specific cases that the evaluator uses the tools of the developer for examination of the implementation representation (e.g. simulation tools, layout viewer).

Layout plans (physical design) describe the organization of the physical components and the signal routing with regard to the process masks and determine the metallisation masks.

Mask plans are necessary for the technological process. The mask plans need only be shown in certain cases if they are necessary for follow-up analyses like vulnerability analyses.

Layouts are required to check the correctness of the implementation of technical and technological properties. The layout indicates the ease with which physical attacks can be mounted (for example, the accessibility of the metallisation layer).

The technical and technological structure of the TOE is to be refined, in order to make an effectiveness analysis possible during the examination of physical attacks on the TOE (e.g. layout information about specific cells might be necessary if they are subject to certain attacks like FIB).

Adjustment of production process parameters will be determined at this stage via consideration of technology specific characteristics and the means of CAD tools.

Sampling of correspondence between TOE design and implementation representation of the TSF
An analysis of the implementation representation is performed by the evaluator with two objectives:

- analyse the correctness of the implementation representation and the TOE design, i.e. traceability of mechanisms implementing the SFR, and the security architectural principles of TSF self-protection, domain isolation, non-bypassability and secure initialization). This analysis uses the mapping between the TOE design and sampled or entire implementation representation in form of schematics/layout shall be provided for this purpose;
- understand the TOE implementation in order to specify potential vulnerabilities and attack paths.

For ADV_IMP.1 the evaluator would selects those parts of the TOE design description that are interesting (especially those crucial for the vulnerability analysis) to verify the implementation representation accurately reflects the description provided in the TOE design description.

In respect of implementation understanding at level ADV_IMP.1 in respect of implementation understanding, the compiling chain from TDS TOE design to the IC the sample of the TOE implementation, shall be documented, provided and their correspondence demonstrated (ADV_IMP.1.2D and ADV_IMP.1.3C). To support the correspondence analysis between TOE design and implementation representation, layout information as well as TOE design deliverables for subsystems and modules should be used. Links between TDS documentation and analogic blocks shall be described and interfaces documented

For example, an FPT_TST.1 (self-tests at start-up and periodically thereafter) requirement may be achieved by environmental sensors, which in turn are represented by security logic and hardware schematics. In this case, operational envelope conditions would need to be traced through the various levels of FPT_TST.1 representation and test evidence.

At level ADV_IMP.2 the required description of relationships between the entire implementation representation and the TOE design shall include justification of relations between modules and the technical and technological structures of the TOE as well as between hardware and firmware parts of the TOE.

2.5.5 TSF internals (ADV_INT)

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

2.5.5.2 Input
The developer shall provide an internal description and justification. Where applicable, the developer shall design and implement the entire TSF such that it has well-structured internals.

2.5.5.3 Requirements
The property “well-structured” depends on the specific technology used for the TOE and typically originates from industry standards for this technology discipline. The CC and the CEM describe criteria to judge about this property for software. For hardware parts of the TSF represented in hardware description language (HDL) the similar criteria apply as for software.

The TSF of security IC is well-structured if the TSF structure provides for minimisation of the complexity of:

- the logical and physical interfaces between modules in a way that the TSF design provides for largely independent modules that avoid unnecessary interactions;
- the functionality in the module, which allows the evaluator as well as the developer to focus only on that functionality which is necessary for SFR enforcement, contributing further to understandability and further lowering the likelihood of design or implementation errors..

The CC characterizes modularity of software as follows: *“software written with a modular design aids in achieving understandability by clarifying what dependencies a module has on other modules (coupling) and by including in a module only tasks that are strongly related to each other (cohesion)”*.

A maximal independence of modules (described according to ADV_TDS.3 or higher) within an IC TOE could be possible if there were no or only minimal physical overlaps and logical dependencies between the individual modules and the interfaces are clearly defined. There are interfaces, which are necessarily complex and cannot be minimized for example buses.

Note the requirements for well-structured internal structure of the TSF do not contradict security IC layout, which hides the structure on the implementation level to increase the barrier for physical attacks (e.g. random module placement, glue logic).

The developer justifies the characteristics used to judge the meaning of “well-structured” used in the internals description and the development process. The evaluator shall examine the justification to determine that it identifies the basis for determining whether the TSF is well-structured (cf. work unit ADV_INT.1-1). The acceptance of the TOE specific criteria for being well-structured should be agreed with the evaluation authority before performing an analysis.

2.6 SECURITY POLICY MODELLING (ADV_SPM)

2.6.1 Objectives

Where applicable, the evaluation of Security Policy Models (SPM) is mainly concerned with formal security policies.

Hence the objective of the requirements is twofold: to determine whether the security policy model clearly and consistently describes the rules and characteristics of the TOE Security Policy and to reinforce the description by formal proof.

The security policy model is considered to structurally formalize the security functionality with sufficient explanatory text as well as to provide the necessary framework to carry out the formal proof of correspondence between the functional specification and the respective policies of the security policy model.

To this end the Security Policy Model of the TOE is informally abstracted from its realization by considering the proposed security requirements of the ST. The informal abstraction is taken to be successful if the TOE's principles (aka rules) turn out to be enforced by its characteristics (see ADV_SPM.1.2C). The purpose of formal methods lies within the enhancement of the rigor of the enforcement; informal arguments are always prone to fallacies, especially if relationships among subjects, objects and operations become more and more involved. In order to minimize the risk of insecure state arrivals the characteristics and rules of the SPM are therefore mapped to their formal counterparts as features and properties within some formal framework, whose rigor and strength can afterwards be used to derive the security properties in terms of theorems.

As structured representations of security policies of the security policy models are therefore used to provide increased assurance that the functional specification corresponds to the security policies of the security policy model, and ultimately to the TOE security functional requirements.

The formal counterpart in terms of principles and features therefore supports the correspondence mappings between the functional specification, the security policy model, and the security policies that are modelled.

2.6.2 Input

Where applicable, the developer shall provide a TOE Security Policy Model.

2.6.3 Requirements

The formal security model is a formal description of the security policy using appropriate formal languages. It is utilized at an abstract level independently from the TOE implementation in hardware or software.

As a guidance on the formality requirements for technical and technological properties (e.g. difficulty of reverse engineering, or operation out of envelope) it may help to model the protection of the IC against unauthorized disclosure of assets, unauthorized use of assets, and unauthorized modification of assets in terms of barriers in combination with the security states of the IC during the different life cycle phases. In many cases a finite state machine seems best suited to satisfy the requirements.

Typically the developer determines the assets beforehand and uses an abstraction appropriate to take care of all desired security functionality. He/she may choose to subdivide the IC operations into security states (e.g. user mode vs. system mode) and state transitions together with subjects acting on objects and causing the state transitions to occur. As the cause of events is proceeding with time the formal system should be at least as strong as to implement mathematical induction to exclude the possibility of insecure state arrivals.

Formal systems appropriate for ADV_SPM.1 include (but are not restricted to) B-Method Isabelle, MetaMath, and VSE II. The formal system shall be agreed with the Evaluation authority of the concrete evaluation process.

The CC do not require that all security functionality be formally modelled, only those policies that can be modelled according to the state of the art. However, Information Flow Control and Access Control almost always are included within the formal model and strong arguments are needed in order to abstain from considering these policies.

2.7 TESTS (CLASS ATE)

The assurance class ATE states testing requirements that demonstrate that the TSF satisfies the TOE security functional requirements. The CC distinguishes four assurance families within this class: Test coverage, test depth, functional tests and independent tests. Note that testing addresses also mechanism defined in ADV_ARC in depth of testing.

2.7.1 Coverage (ATE_COV)

2.7.1.1 Objectives

The objective of these requirements is to determine whether the testing (as documented) is sufficient to establish that the TSF has been systematically tested against the functional specification. Coverage deals with the completeness of the functional tests performed by the developer on the TOE.

2.7.1.2 Input

The developer shall provide evidence (at ATE_COV.1) / an analysis (from ATE_COV.2) of test coverage. This analysis may be part of the test documentation itself or supplied separately.

2.7.1.3 Requirements

The test coverage analysis shall consider the mapping between tests (characterization and production tests) and the TSF as described in the functional specification (security functions). The coverage analysis shall show that all properties of the security functions are covered.

The analysis has to show and justify whether the TOE has been comprehensively tested. Complete coverage of security functions and external interfaces is required for ATE_COV.2. From ATE_COV.3 the analysis has to show that all external interfaces have to be completely tested. For an IC, this can mean for example that the complete instruction set of the CPU with all parameters be covered.

Regarding test coverage, attention must be paid to the inclusion of security functions which result from design or technology. Therefore, evidence has to be provided that technical and technological properties specified in the FSP are covered by tests or other appropriate activities (e.g. layout, mask and chip inspections).

2.7.2 Depth (ATE_DPT)

2.7.2.1 Objectives

The objective of these requirements is to determine the depth of testing. Depth analysis deals with the level of detail to which the developer tests the TOE. Testing of security functions is based upon increasing depth of information derived from analysis of the TSF representations, e.g. whether the developer has tested the TSF against its high-level design.

2.7.2.2 Input

The developer shall provide an analysis (from ATE_DPT.1) of test depth. This analysis may be part of the test documentation itself or supplied separately.

2.7.2.3 Requirements

The depth of testing analysis which is provided for these requirements shall consider the mapping between tests (characterization and production tests) and internal structures of the TSF. Depending on the level of evaluation, this is done at:

- the basic design level (ATE_DPT.1);
- the subsystems and security enforcing modules design level (ATE_DPT.2);
- the subsystems and modules level design level (ATE_DPT.3);
- the subsystems and modules design level and the implementation representation level (ATE_DPT.4).

The application of depth of testing analysis will depend critically on how the terms “subsystems” and ”module” are used for an IC (cf. CC).

All deliverables provided at a certain level (e.g. block diagrams, HDL code, layout documents) should be used for examination of test depth.

For ATE_DPT.3, appropriate depth of testing is achieved when all instructions and branches of the whole logical plan vs. HDL source code, which belong to the SFR enforcing modules, have been tested.

The correctness of the implementation (integration) and the test coverage must also be proven after production (see ATE_FUN). The test vectors must be chosen appropriately, so that they cover the requirements. Analyses should be performed in this way.

The justification for test depth on the implementation level can be done with respect to one of the following items:

- The developer can, if possible, show that he has toggled each junction of a module during testing.
- If, according to the logical plan, the modules or parts of them can only be tested in parallel, the developer must show that all junctions have been toggled at least once via the underlying test vectors.
- If the developer has not taken testability rules into consideration, or the testing of some modules is not immediately possible (the testing of a timer over 24 hours), the circuit cannot be considered 100% tested. In this case, test coverage is achieved if the developer can show that all junctions were achieved via the test vectors and that there are no conditions which compromise security.

The selection of the component ATE_DPT.3 might be sensible to get higher assurance, because the low-level design has been provided anyway and in case of testing technical and technological properties, a low-level design view of tests is sensible.

2.7.3 Functional tests (ATE_FUN)

2.7.3.1 Objectives

The objective of these requirements is to determine whether the developer’s functional testing demonstrates that all TSFI perform as specified.

2.7.3.2 Input

The developer shall test the TSF and document the results. The developer shall provide test documentation. For the conduct of tests, IC data sheets are of particular importance.

2.7.3.3 Requirements

The developer test documentation is required to give details of test plans, goals, and results (actual and expected). Because ATE_FUN.1 is used, the quantity of information that must be presented will vary in accordance with the use of ATE_COV and ATE_DPT.

Tests of individual components of the TOE, or the control of certain technical or technological properties, could only possibly be implemented at a certain time during the manufacturing process or only in test mode, since the respective physical components can be neither logically nor physically accessed after the end of the production of the TOE. This should be considered during test planning and be appropriately documented.

Test plan:
The test plan has to show the objective of the tests, which are to give evidence for the correctness of the logic by means of simulation using the HDL tool and to test the correctness of the implementation. Since a test is a type of quality control, after simulation it must be proven whether the implementation has been successful. Individual tests on the finished IC must show that the implementation of the TSFI and mechanisms is correct, and that the timing requirements are fulfilled. During testing specified functionality or during module testing, binding of modules is of particular note, especially if parallel functionality exists.

Typically, the two main steps in testing a hardware TOE are:

- the ”TOE prototype” tests;
- the acceptance tests performed on each TOE at the end of the production phase.

"TOE prototype" tests are characterisation tests that can be considered to provide evidence for the correct implementation of security enforcing functions. Timing should be considered when testing Hardware TOEs. Tests can also be implemented at the design level with the help of HDL tools (without delays, with estimated loads/ delays and post-layout as appropriate) or in form of special security tests after production using e.g. specific software residing in the TOE (test software within the ROM and being part of the TOE or application test software in the EEPROM not part of the TOE).

Acceptance tests have to confirm and verify the correct operation of the TOE and the components of which it is constructed during its manufacture. Therefore, the evaluators shall check the developer's manufacturing process that it has appropriate acceptance tests implemented. The acceptance testing during production is usually performed using specific hardware mechanisms and commands implemented in the test software on the chip.

The different test environments used for evaluation shall be described within the test plan, e.g.

- ”TOE prototype” tests:
  - characterisation test environment
  - design simulation environment during development
  - security testing environment
- acceptance testing environment during production.

Equipment, which is necessary for a test case, must be specified exactly with all adjustments. This also includes the precise identification of the test libraries for simulation as well as the driver program for the test equipment.

In order to perform or to check the results of characterisation and acceptance tests where specialist test equipment is essential, the evaluator may have to witness and verify the tests rather than personally perform them. This is normally done through a visit to the IC designer/manufacturer.

If the developer would like to do security testing without simulation by means of the HDL tools, all tests must be carried out in real time in order to give evidence that the implementation be correct.

The library of test programs provided by the developer shall contain test programs and tools to enable all tests covered by the test documentation to be repeatable (required for ATE_IND). This may include driver software, among other things, with its associated equipment (tester) required for the testing of the chip. This is also necessary for repeating tests. Other tools that have been used, such as the logic analyser, oscilloscope, debugger, operating system etc. also need to be stated.

A test plan determines the framework of the test cases. In the test plan, the exact specification and scope of the test cases, as well as the documentation describing all input and environment parameters of the IC, are of great importance. These parameters are partially given in data sheets. Therefore the data sheet must be an integral part of the test documentation.

The test cases can vary greatly with analogue and digital circuits.

The test plan should cover all configurations of the IC if specified for the operational environment, e.g. different security states of the IC like test mode and user modes depending on the life cycle phases under evaluation.

Apart from the functional tests under standard conditions, tests (if necessary real time tests) under defined stress conditions (temperature, frequency, voltage, EPROM cycle tests etc.) are also to be planned, since such conditions could arise during the operation of the TOE (comparable with extreme situations for software TOEs, which could lead to run-time errors).

If during operation of the TOE external HW or SW functionality be included dynamically in the functional flow, then the relationship of the external components to the level of the external interface should be tested.

A mapping shall be given between the test cases and TSFI and subsystems, modules or interfaces under test depending on test coverage and depth.

Test parameters must be taken into consideration in the test planning. They could be for example:

- test frequencies with minimum and maximum limits
- voltage supply corresponding to the data sheet
- test temperatures
- test vectors for the selection of the test areas in the IC

Test results:
The way test results are presented by the hardware must be described (e.g. written to a register or certain memory area, sent via an external interface line).

The test results, which will be obtained on special test equipment, must be presented in a form that can be analysed (analogue tests, timing tests).

For test results concerning functionality which run in parallel, the assignment of the results to specific subsystems, modules or security mechanisms is of significance. The dependencies of the results of the tests resulting from parallel processing functionality should be explained.

2.7.4 Independent testing (ATE_IND)

2.7.4.1 Objectives

The objective of these requirements is to determine whether the TOE behaves as specified and to gain confidence in the developer's test results by independently testing a subset of the TSF and by performing a sample of the developer's tests by a party other than the developer (e.g. a third party).

This family adds value by the introduction of tests that are not part of the developer's tests.

2.7.4.2 Input

The developer shall provide the TOE (from ATE_IND.1) and an equivalent set of resources (from ATE_IND.2). The evaluator shall provide test documentation.

2.7.4.3 Requirements

The equivalent set of resources the developer has to provide from ATE_IND.2 may include a separate sample of chips from production, a separate set of test vectors and a separate set of test data necessary for testing.

The evaluator shall provide test documentation. Requirements for test documentation are comparable to those for the developer's test documentation in terms of a test plan, procedures, and expected and actual results.

From the expertise point of view, the evaluator must be able to repeat the developer's tests and perform additional tests. For the conduct of tests, the evaluator needs test vectors, which determine the course of tests. If required, the evaluator must be in the position to use the tools for evaluation applied by the manufacturer. In many cases, owing to tool availability, this will only be possible in the development laboratory or during production by the manufacturer. In these cases, it is sufficient for the evaluator to witness the tests at the manufacturer's site.

Apart from the functional tests under standard conditions, tests (if necessary real time tests) under defined stress conditions (temperature, frequency, voltage, EPROM cycle tests etc.) are also to be performed by the evaluator, since such conditions could arise during the operation of the TOE and may not be extensively tested by the developer.

The additional evaluator tests must be performed at least at the level required by ATE_DPT.

The evaluators must also perform additional tests on a completed IC (final part), because:

- errors could be introduced by technology and may not be detected by logic tests (cf. the ageing process further presented in the document);
- the scattering of security-enforcing and security-relevant parameters cannot be tested via simulation. Such scattering can only be carried out by means of testing several ICs. In order to do this, the evaluator has to select an appropriate sample or rely on the results of the manufacturer's quality tests. For example, the scattering of a mistake in digitalisation can only be detected if several ICs are tested, as assembly of basic components can lead to a timing deviation.

2.8 LIFE CYCLE SUPPORT (CLASS ALC)

Assurance class ALC defines requirements to determine the adequacy of security procedure that the developer uses to protect the TOE development and manufacturing environments. These procedures include the life-cycle model, the configuration management, the handling of security flaws, the tools and the security measures used throughout the TOE development, and the delivery activities.

As defined in CC, 'Development' means here development and production.

2.8.1 CM capabilities (ALC_CMC)

2.8.1.1 Objectives

Configuration Management Capabilities defines the requirements to ensure that the developer has clearly and uniquely identified the TOE using an automated configuration system. The CM system ensures that the TOE is correct and complete during evaluation and before sending to the customer, and prevents any unauthorized modification, addition or deletion of the configuration items.

2.8.1.2 Input

ALC.CMC.4 should be selected.

The developer shall provide CM documentation that includes a CM plan.

2.8.1.3 Requirements

The CM plan shall describe how the CM system is used (ALC_CMC.4.7C), and how the TOE configuration items modification or addition are controlled (ALC_CMC.4.8C) with automated measures (ALC_CMC.4.4C).

The CM system shall be able to automatically generate the TOE (ALC_CMC.4.5C)

The TOE shall be labelled with a unique reference (ALC_CMC.4.1C) and all configuration items shall be uniquely identified. (ALC_CMC.4.3C)

The configuration system and the acceptance procedures should be considered during the whole development and production process of the TOE. They must be in a position also to control the construction plans and hardware parts, in addition to all relevant data files for all development steps. If applicable, various development and production sites are also to be included in this.

The evaluator should ensure that the TOE contains a unique reference such that it is possible to distinguish between different versions of the TOE. The TOE may provide a method by which it can be easily identified. For hardware TOEs this may be a part number physically stamped on the TOE. Furthermore, each TOE mask layer may be physically identifiable by use of any kind of identifiers.

However, in certain cases it can be necessary, in order to make an attack more difficult, to mark the ICs (or the chips held within the ICs) with non-visible logos or IDs. In such cases, the manufacturer must, however, find a suitably hidden possibility for the label on the TOE, such as for example, in a non-deletable area of memory with access for authorized users only.

The evaluators work unit at ACM_CMC.4.5C to examine the TOE generation procedures has the objective to get evidence about the effectiveness of the configuration control system with regard to the various versions and changes to the TOE. It has to be shown that the configuration control system supports the generation process to help reduce the probability of human error. Thus, the generation process makes use of the appropriate design tools (HDL / CAD tools). This should be described.

In the case of an IC TOE comprised of hardware and software (e.g. Test-ROM software, operating system, smart card application software depending on the specific scope of a TOE) there is likely to be a distinction between the hardware and software configuration control. There is an additional requirement to bring together the right hardware-software pair, which means that the right mask must be used. This in turn means that the configuration control system must be able to administrate hardware-software pairs comprising a TOE. Because all masks are created from mask data files it has to be ensured that masks are created from their correct software image.

Depending on the scope of the TOE, bringing together the right hardware-software pair can partly be an aspect of the TOE configuration management or of the delivery procedures (see ALC-DEL).

It is not possible to relate this work unit for generation of the TOE directly to the technological process of a customer specific IC, because the process cannot be repeated for the benefit of the evaluator. In this case, the evaluator must, however, audit the configuration control in the technological process in order to guarantee that the correct masks, which belong to a particular version of the TOE, are used and organizational measures are effective in the process.

In the case of programmable standard ICs (PLD, FPGA), in which the hardware configuration is programmed via firmware, the work unit may be supported by programming a new IC. The evaluator then conducts comparison tests of the functions of the newly produced IC with the original TOE (to be equated with a ‘file compare’ for a re-built software TOE).

2.8.2 CM scope (ALC_CMS)

2.8.2.1 Objectives

The objective of these requirements is to identify the items to be included in the configuration list and hence placed under the CM requirements as per ALC_CMC.

2.8.2.2 Input

ALC_CMS.4 or ALC_CMS.5 should be selected..

The developer shall provide the TOE configuration list.

2.8.2.3 Requirements

The configuration list shall include the TOE itself, the evaluation evidences required by the SARs, the parts that comprise the TOE, the implementation representation, the security flaw reports and resolution status (ALC-CMS.4.1C). Additionally, development tools and related information shall be considered in the configuration list (ALC_CMS.5.1C).

The developer performs configuration management on the TOE implementation representation (hardware schematics/layouts), design documentation, tests, user and administrator guidance, the configuration management documentation and security flaws with flaws resolution status.

Configuration management shall be in place for IC design (schematics, layout) as well as IC proprietary dedicated software (source code, documentation). All source files necessary for the generation of the TOE as well as the identification of the set of masks necessary for the manufacturing of the TOE have to be included. For masks, this includes unique mask identifier, as well as the version number or revision number of each layer.

Because the configuration control system must be able to administrate hardware-software pairs as part of the TOE (see ALC_CMC), there must be evidence at ALC_CMS which specific hardware-software pair is used for the TOE.

The identification and listing of modules of the design in the configuration list seems to be difficult to apply to ICs. Since, within the framework of the design, functional blocks can be taken out of a developer's HDL or CAD library and out of the IC manufacturer's technology library (lists of the technology parameters). At the very least the libraries as a whole that are used, together with the possible parameters that are used, must be clearly identified if individual library components do not have their own identifier.

Test information covered by CM shall include all of the parts which are necessary for the generation and testing of the TOE. That includes all test equipment, libraries, and the list of test vectors used during testing as well as the set of tests including test data and results.

ALC_CMS.4.3C and ALC_CMS.5.3C requires the configuration list to indicate the developer of the item. As defined in CC3., "Developer" here refers to the organisation responsible for the development of the item. That is e.g. any developer of an additional software-part (library) to be used by the TOE (see CC). The intention is to give evidence on the origin of parts of the TOE provided from external suppliers or if the development is done within different organizations.

2.8.3 Delivery (ALC_DEL)

The assurance family ALC_DEL defines requirements for the measures, procedures, and standards concerned with secure delivery, of the TOE, ensuring that the security protection offered by the TOE is not compromised during transfer.

2.8.3.1 Objectives

The objective of these requirements is to determine whether the delivery procedures are documented and maintain integrity and the detection of modification or substitution of the TOE when distributing the TOE to the user's site. It includes special procedures or operations required to demonstrate the authenticity of the delivered TOE. Such procedures and measures are the basis for ensuring that the security protection offered by the TOE is not compromised during transfer.

2.8.3.2 Input

The developer shall provide delivery procedures of the TOE or parts of it to the user. The procedures shall be described and used.

2.8.3.3 Requirements

Requirements address delivery to Users. As per CC, transportation from subcontractors to developers or between different sites is not considered, but in ALC_DVS.

The examination of the delivery process to determine that the delivery procedures are used is normally done during site inspections. Traceability of what has been delivered by who to whom is verified. A particular attention is placed on "parallel deliveries" such as samples for quality inspection, scrapped samples, screened processes.

If applicable, delivery procedures approved by the national certification body should be followed.

Note that a specific authentication mechanism (e.g. fab-key, transport key) may be used to protect the delivery of the chip from the chip manufacturer to the Card Manufacturer/personalization centre.

For ICs the security state of the chip (test mode, user mode) during delivery is of importance. Functionality for the deactivation of test hardware or for the transition from test mode or installation mode to user mode/operational mode is important in the context of vulnerability analysis and the authenticity of the delivered TOE. It should be mentioned here with reference to the description in AGD_PRE.

The TOE manufacturing and delivery process shall include production tests that ensure the correct function of each TOE example delivered to the costumer. The results of these tests must be documented.

2.8.4 Development security (ALC_DVS)

2.8.4.1 Objectives

The objective of these requirements is to determine whether the development and manufacturing environment procedures are adequate to provide the confidentiality and integrity of the TOE design, implementation and production that is necessary to ensure that secure operation of the TOE is not compromised.

2.8.4.2 Input

The developer shall provide development security documentation.

2.8.4.3 Requirements

ALC_DVS.1.1C requires that documentation describes all security procedures used to protect the TOE during development. ALC_DVS.2.2C additionally requires that the documentation justifies that security measures provide the necessary level of protection.

Note that ALC_DVS gives the possibility to define the level of security in Confidentiality and Integrity. Referring to CC "*It is recognised that confidentiality may not always be an issue for the protection of the TOE in its development environment. The use of the word "necessary" allows for the selection of appropriate safeguards*." For instance for 'open source software', no confidentiality is required.

During the life-cycle of the hardware TOE, development and manufacturing security procedures are examined. All relevant development and production sites of the TOE must be taken into consideration so that the security requirements are valid for all life cycle phases until the final delivery of the TOE within the scope of the evaluation. This is of particular importance because the requirements on the technology are finally only realized during the production of the ICs, including tests for correct function of all TOE examples delivered to the costumer. The examination includes all the steps of the development and manufacturing process; the following sites are concerned:

- development of dedicated software and hardware (design centre),
- site for creating the image of the application software (if applicable),
- reticles manufacturer (mask manufacturer),
- manufacturing (fab site),
- testing (test site),
- packaging (microassembly and testing) depending on the scope of the TOE.

In specific cases there may be a separate design centre for certain cells that are not TOE specific (e.g. standard CPU cell, standard memory cell). For the requirement ALC_DVS.1, there may be enough evidence that these pre-defined cells (not security enforcing but possibly security supporting) are functionally correct and integer if appropriate delivery procedures, tests and confidentiality agreements are in place. In this case this design centre would not have to be considered under ALC_DVS. The site of the mask manufacturer can be treated accordingly.

All the security operational procedures are being checked. The examination is normally done during site inspections. Subcontractor procedures are also checked. It should be noted that evaluator access to manufacturing site, specialist personnel and tools would be required to support these evaluation activities.

The Procedures shall include the following types:

- physical (site security: access controls);
- procedural (granting of access to the development tools, revocation of access, transfer of protected material, admitting and escorting visitors, development security policy ...);
- personnel (screening process for new development staff ...);

- IT security measures (Identification and authentication, access control, archiving, audit, networking, firewalls ...).

Further sensitive areas within the sites mentioned above might be:

- process control (integration of the circuits onto silicon);
- product engineering (fault analysis with regard to the process);
- quality control for security functionality;
- storage / delivery.

The TOE is present in the various stages of development and production in various different physical shapes. The integrity of the layout masks is of particular significance. Secure delivery will support this aspect (see ALC_DEL).

Physical, procedural, personnel and other measures necessary for the realization of the TOE's security properties, as given in the Security Target, will be transposed in the development and production and will have an effect on the security in the operational phase of the TOE. These measures are also to be documented and examined. Measures in the test phase, as well as the assembly phases if applicable, and measures for the management of the manufacturing process are particularly important.

In comparison, compilation of a software TOE takes place with fixed options uniquely in the development environment (prototype and master copy). The series production of the software is simply a process of copying, in which the integrity aspects of the copy with respect to the master copy play a role.

With respect to IC TOEs, drawings and associated data files will be created within the framework of the development. The IC production of the prototype as well as the series is essentially more complex than a copying process in the case of software and is variable through a multitude of process parameters, which are potentially manipulated by personnel.

The test phase during IC production is of particular importance, since in this phase an IC is already completely physically available, but, for example, internal IC structures are, however, adjustable or may be compromised via a test mode, which is still activated.

Measures, which are taken, in order to mark (ink) the faulty dice on the wafer and to sort out faulty TOEs (final parts), including which criteria to select, can be of importance. Measures for the destruction of defective parts should then be described.

2.8.5 Flaw remediation (ALC_FLR)

2.8.5.1 Objectives

Flaw remediation ensures that flaws discovered by TOE consumers will be tracked and corrected while the TOE is supported by the developer. While future compliance with the flaw remediation requirements cannot be determined when a TOE is evaluated, it is possible to evaluate the procedures and policies that a developer has in place to track and repair flaws, and to distribute the repairs to consumers.

2.8.5.2 Input

The developer shall document the flaw remediation procedures.

2.8.5.3 Requirements

ALC_FLR.2 should be selected.

Aspects of flaw remediation might be combined with the use of an evaluation maintenance programme.

2.9 LIFE CYCLE DEFINITION (ALC_LCD)

2.9.1 General Remarks

2.9.1.1 Objectives

The objective of these requirements is to determine whether the developer has used a documented model of the TOE life-cycle for development and maintenance.

Life cycle definition establishes that the engineering practices used by a developer to produce the TOE include the considerations and activities identified in the development process and operational support requirements.

Confidence in the correspondence between the requirements and the TOE is greater when security analysis and the production of evidence are done on a regular basis as an integral part of the development process and operational support activities. It is not the intent of this component to dictate any specific development process.

2.9.1.2 Input

The developer should provide a life-cycle definition.

2.9.1.3 Requirements

The description of the model should include information on the procedures, tools and techniques used by the developer for development and maintenance of the TOE.

An example of the life-cycle model is detailed in the Security IC protection Profile [BSI-CC-PP-0084-2014]. This model shall be refined by the developer.

A basic description is required at ALC_LCD.1, whilst at ALC_LCD.2 the life-cycle model used shall be based on a measurable life-cycle model, i.e. one that has been approved by academic experts or standards' bodies.

2.9.2 Tools and techniques (ALC_TAT)

2.9.2.1 Objectives

The objective of these requirements is to determine whether the developer has used well-defined tools to develop, analyse and implement the TOE (e.g. programming languages or computer-aided design (CAD) systems) that yield consistent and predictable results.

It includes requirements concerning the development tools and implementation dependent options of those tools.

2.9.2.2 Input

The developer should provide development tools' documentation being used for the TOE.

2.9.2.3 Requirements

The evaluation aspect for tools and techniques is applicable to software as well as hardware TOEs.

When considering tools and techniques for software TOEs, it is a question of getting evidence whether the development tools which have been used are unambiguously and well defined and documented, and whether all options of the tools have been documented. From ALC_TAT.2 implementation standards have to be applied. The objective here, apart from a higher assurance into the correct implementation of the TOE, is also to ensure repeatability of the construction of the TOE. The required confirmation of the evaluator that the implementation standards have been applied may require visiting all relevant sites. Therefore, it should be noted that evaluator access to manufacturing site, specialist personnel and tools would be required to support these evaluation activities.

For the dedicated software of the IC, this corresponds to the software development tools.

The use of the development tools shall be documented. This includes in particular the description of the compiling chain for the software source code (if applicable) and the synthesis chain for HDL. All options shall be documented and parameters shall be clearly identified. The evaluator shall verify this tools' chain, usually during site inspection.

In order to achieve the objective of ALC_TAT for a hardware TOE, it is necessary to document and test the hardware description languages (HDL), representation elements (graphical logic elements) and tools (e.g. HDL-compiler, simulation tools, and CAD tools) used in the hardware. Additionally supporting libraries shall be considered. A clear and precise definition of all elements and the options used for the TOE is important.

In the case of software, various compilers can create different object codes even with the same functionality of the TOE (i.e. with the same logical design). Functionality is defined by the processor commands and the compiler options that are used.

In the event that microchip ICs have different masks, differing physical implementations can arise as a result of different technologies, even though functionality may be the same (i.e. with the same logical design at the level of the circuit diagram). Functionality is defined finally only by means of the cell structure which has been implemented in the silicon. As a result of this, the technology used for the implementation of the chip has to be specified. At high assurance levels, the parameters of the technology used have to be documented.

For the purpose of clarification, the following table shows the development processes of hardware ICs and software:

Table 2 - Development processes of hardware ICs and software

Software	Hardware
Program text input via the editor for the creation of the source file. Syntax and semantics of the input language will be determined via the compiler.	Creation of the logical plan through graphic input via a CAD tool or text input via a HDL-Editor. Syntax and semantics of the graphics symbols are determined via the CAD tool and the technology. For the modelling and the simulation of the circuit design a HDL will be used.
In order to construct a functional software TOE, several steps are required: Determining the compiler and linker adjustments, in order, for example, to realise certain optimization possibilities. Compiling and linking of the source files to a program which is executable from a processor during its run time (object files as program data files, run time library, program code for a hardware memory). Testing and de-bugging within the framework of the compilation of individual source files as well as the whole TOE.	In order to construct a functioning IC from the design, several steps are needed: The construction of a netlist out of the logical plans and the synthesis of the gate structure as well as the test structure. The simulation of this logic at the gate level and at the level of the layout using timing defaults. The construction of the layout and the masks. The manufacture of the microchip from this masks after several process steps, which are dependent on the semiconductor technology used (e.g. 0.8_ìm CMOS-, BiCMOS- or Bipolar technology).

A programming language used is based on the features of a compiler, interpreter or assembler, while the logic which has been constructed using an HDL is finally only available after the conclusion of the technological process. Therefore, this assurance aspect is to be understood in the sense of ”Tools, Techniques and Technology”.

2.10 GUIDANCE DOCUMENTS (CLASS AGD)

The Common Criteria assurance components of the families AGD_OPE (Operational user guidance) and AGD_PRE (Preparative user guidance) “*describe all relevant aspects for the secure application of the TOE*.“

2.10.1 Operational User Guidance (AGD_OPE)

2.10.1.1 Objectives

The Operational User Guidance documents should provide only the information which is necessary for using the TOE. Depending on the recipient of that guidance documentation Operational and Preparative User Guidance can be given in the same document.

The TOE serves as a platform for the Security IC Embedded Software. Therefore the role of the developer of the Security IC Embedded Software is the main focus of the guidance.

2.10.1.2 Input

The developer shall provide guidance documentation. The datasheet of the IC could be considered to support these requirements.

2.10.1.3 Requirements

If the TOE provides security functionality which can or need to be administrated by the Security IC Embedded Software or if the IC Dedicated Support Software provides additional services, these aspects must be described in Guidance.

Most of the security functionality will already be effective before TOE Delivery. However, guidance to determine the behaviour of security functionality, to disable, to enable or to modify the behaviour of security functionality must be given if a configuration is possible after TOE Delivery (that means either by the Developer of the Security IC Embedded Software or by the Composite Product Manufacturer). This guidance is delivered by the TOE Manufacturer.

If the Composite Product (with the TOE as a major element) is used in a terminal where communication is performed through the interface provided by the TOE in combination with the Security IC Embedded Software then Guidance must be given to the developer of the terminal. This is information about the physical characteristics of the device, the interface and standard protocols if implemented by the TOE.

Guidance documents must not contain security relevant details which are not necessary for the usage or administration of the security functionality of the TOE.

2.10.2 Preparative User Guidance (AGD_PRE)

2.10.2.1 Objectives

Preparative user guidance is intended to be used by those persons responsible for secure acceptance and installation of the TOE as well as the secure preparation of the operational environment in a correct manner for maximum security.

2.10.2.2 Input

The Family AGD_PRE addresses the activities of the delivery acceptance procedures. For the hardware platform this comprises procedures that can be applied to identify the TOE and eventually to verify the authenticity of that part of the TOE.

2.10.2.3 Requirements

The TOE may be configured after production before the Composite Product is delivered to the consumer. In this case, these configuration aspects have to be considered.

The preparation may include e.g. the download of Security IC Embedded Software. If the TOE includes software that is delivered separately, the preparation includes integration of the IC Dedicated Support Software. The preparation also includes the configuration of the TOE according to the options described in the Security Target that can be changed after TOE delivery. The guidance documentation shall describe all relevant procedures.

2.11 VULNERABILITY ASSESSMENT (CLASS AVA)

The AVA: Vulnerability assessment class addresses the possibility of exploitable vulnerabilities introduced in the development or the operation of the TOE. It consists only of one family Vulnerability analysis AVA_VAN, which comprises all aspects of vulnerability assessment.

2.11.1 Vulnerability analysis (AVA_VAN)

2.11.1.1 Objectives

Vulnerability analysis is an assessment to determine whether potential vulnerabilities identified, during the evaluation of the development and anticipated operation of the TOE or by other methods (e.g. by flaw hypotheses or quantitative or statistical analysis of the security behaviour of the underlying security mechanisms), could allow attackers to violate the SFRs.

The levelling of the components of the AVA_VAN family is based on an increasing rigor of vulnerability analysis by the evaluator and increased levels of attack potential required by an attacker to identify and exploit the potential vulnerabilities.

2.11.1.2 Input

The vulnerability analysis is subject of the evaluator. The developer has to provide the TOE for penetration testing.

The evaluators use all information they got and consider all potential vulnerabilities encountered during the conduct of other evaluation activities. The vulnerability analysis is based upon analysis performed by the evaluator, and is supported by evaluator testing.

2.11.1.3 Requirements

General aspects of vulnerability analysis

The CEM identifies three main factors in performing a vulnerability analysis, namely:

a) the identification of potential vulnerabilities;
b) assessment to determine whether the identified potential vulnerabilities could allow an attacker with the relevant attack potential to violate the SFRs.
c) penetration testing to determine whether the identified potential vulnerabilities are exploitable in the operational environment of the TOE.

The assessment of TOE resistance against attack is subject of the analysis. Therefore, the evaluator can use the public available documentation on potential vulnerabilities and potentially flaws only as a basis for the analysis. The evaluator must consider also modifications and improvements of the known attacks to verify the resistance of the TOE. The detailed information on the TOE design can support the adaptation of attacks and support the analysis of the resistance against specific attack steps.

This work requires expertise and equipment as outlined in the state-of-the-art document supporting the EUCC scheme MINIMUM ITSEF REQUIREMENTS FOR SECURITY EVALUATIONS OF SMART CARDS AND SIMILAR DEVICES. At least the evaluator must perform penetration tests to such an extent that the overall rating of the attack path can be proven by the provided test results.

Development and operational vulnerabilities
Vulnerabilities can be introduced in both the construction of the mechanism itself, as well as through the production of technical and technological measures intended to counter threats. The effectiveness of the functionality depends on the technology used in the implementation phase. This must be taken into consideration in the vulnerability analysis.

Development of vulnerabilities takes advantage of some property of the TOE which was introduced during its development, e.g. defeating the TSF self-protection through tampering, direct attack or monitoring of the TSF, defeating the TSF domain separation through monitoring or direct attack the TSF, or defeating non-bypassability through circumventing (bypassing) the TSF.

Operational vulnerabilities take advantage of weaknesses in non-technical countermeasures to violate the TOE SFRs, e.g. misuse or incorrect configuration. Any ways in which the SFR may be deactivated, bypassed or corrupted should be analysed and assessed by the evaluator. This analysis must provide arguments as to why the vulnerability cannot be exploited within the TOE’s environment.

The operational vulnerabilities are also to be considered in the context of the use of the chip by an operating system or an application developer. For instance, the security measures which should be taken in the application development and which influence the operation of the IC could possibly be exploited by an attacker (e.g. demands on external cabling, external technical parameters, or confidentiality measures).

The process of ageing the IC should be taken into consideration during vulnerability analysis. So, for example, the vulnerabilities of an IC TOE can lie in the semiconductor technology. E2PROM cells only withstand a restricted number of program cycles. The limitation of the number of possible delete and write cycles of a cell is an inherent vulnerability, which an attacker could possibly exploit. This kind of technologically based vulnerability analysis is new in contrast with software TOEs, and requires a vulnerability analysis relating to the technology and its implementation.

Typical vulnerabilities
For smartcards and similar devices, the state-of-the-art document supporting the EUCC scheme APPLICATION OF ATTACK POTENTIAL TO SMARTCARDS AND SIMILAR DEVICES is considered as basis for search of potential vulnerabilities. Other activities will be performed according to standard CC practice as documented in the CC, the CEM, and other EUCC documentation.

Hardware TOEs can be subject to vulnerabilities which can be exploited by physical tampering of the TOE. With respect to attacks which physically modify the internal technical structures of the TOE, it is a question of an indirect attack which needs to be examined in the context of a vulnerability analysis, since security features may be bypassed and therefore may lose their effectiveness. Such tampering could circumvent the effectiveness of other security mechanisms. Additionally, the binding of distinct components realized in mechanisms has to be taken into consideration. This aspect must be considered during vulnerability assessment and penetration testing.

Binding aspects should be considered during vulnerability analysis, owing to vulnerabilities, which may arise from problems in binding if the TOE’s security functions are not mutually supportive or do not provide an integrated and effective whole. In particular, it addresses the issue of indirect attack against the IC, e.g.:

- physical connections between physical components in the form of signal paths and circuits;
- physical connections between physical components because of the layout (i.e. that information on the technical and technological implementation needs to have some influence in the analysis);
- dynamic interweaving in the timing behaviour of individual security functions or mechanisms;
- influence on binding through the setting of external signals on the microchip.

Some hardware security mechanisms are only effective in combination with additional software countermeasures in a composite product. Therefore additional vulnerability analysis of the security mechanism of the composite product is necessary as defined by the state-of-the-art document COMPOSITE PRODUCT EVALUATION FOR SMART CARDS

AND SIMILAR DEVICES. These hardware security mechanisms must be evaluated to such an extent that the composite evaluator is able to assess the combination of hardware and software. A related description must be included in the ETR for composition. Open samples may be used by the composite evaluator to tune the test bench for the composite product.

Penetration testing
The penetration testing consists of an analysis of the TOE based on potential vulnerabilities and potentially flaws that are available from the public domain and scheme specific documentation.

Attack potential quotation
One key aspect of the vulnerability analysis requirements is the notion of resistance to attack posed by attackers who have a particular attack potential (basic, enhanced-basic, moderate or high attack potential). The attack potential considered for a TOE evaluation is pre-defined by selection of a certain AVA_VAN assurance component in the ST.

The resistance to attacks to be provided by the TOE depend on:

- the assets to be protected and the perceived risk of compromise of those assets;
- the intended operational environment determining the perceived attacks and the factors like windows of opportunity;
- the perceived market requirements under consideration of costs for development, manufacturing and certification.

E.g. subscriber identity modules (SIM) stores the service-subscriber key used to get access to mobile phone networks. The value of this service for an attacker cloning the SIM under risk for being detected and blocked by the network operator may be low. In this case basic resistance of the SIM against cloning may be appropriate. In other cases like pay TV storing cryptographic keys used by many subscribers in undirected communication may require high resistance against attacks compromising such key.

Therefore, for security IC or smartcard evaluations, the selection of AVA_VAN.5 should be considered.

State-of-the-art document APPLICATION OF ATTACK POTENTIAL TO SMARTCARDS AND SIMILAR DEVICES, gives guidance for the rating of the attack potential necessary to perform specific attacks. It is a interpretation for security IC based on the CEM, considering the specific technology and operational environment as well as the need of information about the security during the operation usage phase like separate rating of identification and operation for risk management. Rating provides a measure of the weakest path required to determine IC secrets or tamper with the IC. In practice, this is likely to be the sum of various work functions (e.g. remove protective barrier, determine IC layout, decrypt data or extract EEPROM contents).

Different methods can be used to prove the rating:

- Performing penetration tests up to a specific step of the attack path that proves the possibility but require considerable additional effort for the exploitation. For example power glitches are performed to such an extent that the resulting faults are reproducible and the effect of the faults can be determined or reverse engineering is performed to such an extent that a specific part of the circuit can be verified without usage of design information.
- Comparing results of penetration tests with a test setup not available to an attacker with the results of penetration test with a test setup available to an attacker. For example a SPA/DPA analysis is performed with chosen values or specific configurations and compared with random values and the configuration under evaluation.
- Running automated tests for the same time period that is used in the rating. For such tests the functionality of the test setup must be verified beforehand. The results of such tests may be partly successful however they shall not include any results that show a direct vulnerability. For example each semiconductor is sensitive to light attacks. Therefore the complete surface of the device must be scanned to check for faults that may provide indications for further successful attacks.

According to the CEM, the attack potential calculation does not distinguish any more between the identification phase and the exploitation phase but within the community for smartcards and similar devices, the risk management performed by the user of CC certificates required clearly to have a distinction between the cost of “identification” (definition of the attack) and the cost of “exploitation” (e.g. once a script is published on the web). Therefore this distinction is kept in the state-of-the-art document APPLICATION OF ATTACK POTENTIAL TO SMARTCARDS AND SIMILAR DEVICES when calculating attack potential for evaluation of smartcards and similar devices. Although the distinction between identification and exploitation is essential for this type of products evaluation to understand and document the attack path, the final sum of attack potential is calculated by adding the points of the two phases, as both phases build the complete attack.

When using the rating tables from the state-of-the-art document APPLICATION OF ATTACK POTENTIAL TO SMARTCARDS AND SIMILAR DEVICES or the CEM, justification shall be provided why the assertions or assumptions supporting the analysis are valid (e.g. why a certain level of expertise or certain equipment for an attack is applicable and no less).

3 GLOSSARY

BiCMOS	Bipolar Complementary Metal Oxide Semiconductor, specific semiconductor technology
CAD	Computer Aided Design
CMOS	Complementary Metal Oxide Semiconductor, specific semiconductor technology
Die	individual IC on a wafer (plural ”dice”)
EPROM	Erasable Programmable Read Only Memory
E2PROM	Electrically Erasable Programmable Read Only Memory
HDL	Hardware Description Language
HW	Hardware
IC	Integrated Circuit, integrated electronic circuits in a microchip
SW	Software
Wafer	Silicon slice for chip production

ABOUT ENISA

The mission of the European Union Agency for Cybersecurity (ENISA) is to achieve a high common level of cybersecurity across the Union, by actively supporting Member States, Union institutions, bodies, offices and agencies in improving cybersecurity. We contribute to policy development and implementation, support capacity building and preparedness, facilitate operational cooperation at Union level, enhance the trustworthiness of ICT products, services and processes by rolling out cybersecurity certification schemes, enable knowledge sharing, research, innovation and awareness building, whilst developing cross-border communities. Our goal is to strengthen trust in the connected economy, boost resilience of the Union's infrastructure and services and keep our society cyber secure. More information about ENISA and its work can be found at www.enisa.europa.eu.

ENISA
European Union Agency for Cybersecurity

Athens Office
1 Vasilissis Sofias Str
151 24 Marousi, Attiki, Greece

Heraklion office
95 Nikolaou Plastira
700 13 Vassilika Vouton, Heraklion, Greece

enisa.europa.eu