# D11.2

## Cybersecurity compliant development processes

| | |
|---|---|
| **Project number** | 830892 |
| **Project acronym** | SPARTA |
| **Project title** | Strategic programs for advanced research and technology in Europe |
| **Start date of the project** | 1st February, 2019 |
| **Duration** | 36 months |
| **Programme** | H2020-SU-ICT-2018-2020 |

| | |
|---|---|
| **Deliverable type** | Report |
| **Deliverable reference number** | SU-ICT-03-830892 / D11.2 / V1.0 |
| **Work package contributing to the deliverable** | WP11 |
| **Due date** | July 2020 – M18 |
| **Actual submission date** | 31st July, 2020 |

| | |
|---|---|
| **Responsible organisation** | SAP |
| **Editor** | Volkmar Lotz |
| **Dissemination level** | PU |
| **Revision** | V1.0 |

| | |
|---|---|
| **Abstract** | This deliverable describes an alternative approach to product and service certification that is suitable for agile and dynamic development environments. Instead of analysing the product itself, it is based on assessing the processes, tools and methodologies that form the secure development life cycle at the vendor's organisation, as well as on evidence that these processes are applied properly for the development of the product or service at hand. Compared to product-centric security certification approaches, the process-centric methodology has the potential to scale and cope with agility. |
| **Keywords** | Certification, secure development life cycle, process-centric, agile development, scalability |

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 830892.

**Editor**

Volkmar Lotz (SAP)

**Contributors** (ordered according to beneficiary numbers)

Volkmar Lotz (SAP)

**Reviewers** (ordered according to beneficiary numbers)

Giuseppe Bianchi (CNIT)

Jerôme Jacob (SMILE)

## Disclaimer

# Executive Summary

This deliverable describes an alternative approach to product and service certification that is suitable for agile and dynamic development environments. Instead of analysing the product itself, it is based on assessing the processes, tools and methodologies that form the secure development life cycle at the vendor's organisation, as well as on evidence that these processes are applied properly for the development of the product or service at hand. Compared to product-centric security certification approaches, the process-centric methodology has the potential to scale and cope with agility.

The document does not describe a full certification scheme but focuses on the basic principles and components of a process-based approach to software product and service certification. First, it analyses the requirements for software product and service certification that need to be met if security certification should be attractive for a commercial market: coping with complexity, scalability and agility of today's software, supporting a risk-based approach and keeping the costs of certification manageable. Second, it introduces the basic constituents of a certification approach that meets the requirements: a secure development life cycle (SDLC) and its process elements that can be tailored to individual products or services following a risk assessment, ensuring a high security quality by aligning with best security practices and requiring validation of the application of the process, a catalogue of process elements that can be instantiated and combined to a product-specific SDLC and meaningful combinations of process elements, their instantiations and their validation activities into assurance levels.

The proposal for a process-based approach for security certification is analysed in detail against the requirements stated, the state-of-the-art in secure development, deployment and operations practices, and for their support of the security objectives stated in the European Cybersecurity Act. It turns out that it is designed in a way that allows to meet the requirements, and, hence, is a good basis for a security certification scheme that is successful in a commercial environment, in terms of ease of adoption and economic viability. The approach itself is scalable and can be tuned from very basic inexpensive checks of process definition and application to rigorous evaluation. This is demonstrated by showing how the approach can be extended to support full common-Criteria-style product-centric evaluations and certifications.

# Table of Content

# List of Figures



# List of Tables

# Chapter 1 Introduction

*This chapter motivates the need for alternative certification approaches for products and services compared to existing ones like Common Criteria. It gives some background on the role of certification in cybersecurity (e.g., by reference to the Cybersecurity Act), emphasises on the voluntary approach carried by the EU-CSA, and argues that cybersecurity can benefit from certification becoming a routine activity not only for critical systems, but for any type of commercial system. To facilitate the latter, certification must be economically and technically feasible while at the same time providing strong security claims.*

The continued digital transformation of business, government, society and private life increases both the dependency on ICT infrastructures and systems and the speed in which new digital solutions are made available on the market. Infrastructure technologies like cloud computing or 5G mobile communication allow the fast processing of vast amounts of data, scaling resources and features to the actual demand of customers. This leads to highly dynamic and flexible systems that can immediately respond to changed requirements and contexts.

The characteristics of the new technologies have dramatically changed the way software systems are built today. Agile methods with extremely short release cycles dominate commercial software development and lead to seamless integration of development activities and system operations ("DevOps") and a continuous stream of new releases of systems and services through automated build processes ("Continuous Deployment / Continuous Integration – CD/CI") [1]. Software vendors and cloud providers focus their own development investments on core functionalities, while consuming the remaining software from 3rd party vendors including open source software. The price to pay for the required agility and flexibility is an increasingly complex supply chain with rich dependencies as well as an increased difficulty to analyze and assess the properties of such systems.

The digital transformation can only show its full potential, if cybersecurity risks can be managed, and if customers, consumers and users can place trust in the underlying technology. Certification is a well-established traditional means to define and formalize desired properties and behaviors or best practices to achieve them – by establishing criteria – and to gain confidence about the validity of such properties and behaviors – by evaluation of a system or service against the criteria. This role of certification for cybersecurity has been exemplified by the long history of certification schemes – ranging back to the US Orange Book of 1983 [2] –, by the many certification schemes that have been established since then – a report by ECSO [3] counts almost 100 of them –, and by its prominent role in the cybersecurity strategy of the European Commission [4]. The European Cybersecurity Act [5] (EU-CSA for short) became effective in June 2019 and establishes the European Cybersecurity Certification Framework, targeting the security of products, services and processes and under which the European Cybersecurity Agency (ENISA) is expected to propose several harmonized schemes in the coming years, including a scheme for cloud services which is currently under preparation.

In this deliverable, we focus on cybersecurity certification of commercial products and services that drive the digital transformation of economy and society and exhibit the characteristics described above: highly dynamic, short release cycles and relying on a complex supply chain. We call such products or services "modern commercial" for short. With a focus on the criteria, we motivate that schemes targeting the evaluation of single products and services with the aim of understanding and demonstrating how the security mechanisms of the particular product or service meet their security requirements do not scale to modern commercial systems, and propose an alternative approach based on evaluating *how* a product or service has been developed and is operated rather than *what* has actually been developed and deployed.

We are convinced that certification is a powerful instrument in the cybersecurity arsenal and that the security of commercial systems (distinguished from critical infrastructures and mission-critical systems) can highly benefit from certification. Following this conviction, our goal is to provide the foundation for a cybersecurity certification scheme that appeals to many software vendors in the commercial space, including cloud service providers, and that is not prohibitive in terms of effort, timeline and costs required to undergo a certification of the vendors software products or software services. Economic viability is essential to the success of certification in the commercial space: while in regulated markets the regulator can set the conditions and mandate certification of products and services, the commercial space is sensitive to the balance of required efforts and the market rewards resulting from them (cf. the “Market of Lemons” [6]).

To enable the use of certification as a facilitator of improved cybersecurity and to prepare the grounds for certification becoming a routine activity in all sorts of commercial spaces for software products and services, the EU-CSA is following a voluntary approach to cybersecurity certification (with the option of regulated sectors imposing mandatory certification for their respective scope). If market mechanisms decide about the attractiveness and relevance of cybersecurity certification, the respective schemes need to be suitable for the products and services targeted, and certification must be achievable with manageable effort (in all dimensions including resources, time and cost) and meaningful claims. Only then the benefits for the vendors of certified products and services – a uniform demonstration of their security qualities based on agreed best practice criteria avoiding to respond to as many different sets of requirements as there are customers – as well as for the consumers of such products and services – having aligned and comparable statements about the security of their purchases – can be realized.

For modern commercial systems, products and services showing the characteristics described above, this means that the certification schemes must be as agile and flexible as the systems themselves. The current dominant approaches to cybersecurity certification do not scale: they are either too much focused on the product or service version or instance (like the Common Criteria which cannot catch up with the speed of CI/CD) or on security management aspects (like ISO27001 or cloud security schemes). In this deliverable, we therefore propose an approach emphasizing on the secure software development lifecycle (SDLC) and its process elements, based on the observation that these are much less dynamic than their outcomes, but still establish best cybersecurity practices that are reflected in the product’s or service’s security qualities.

This deliverable is organized as follows. In Section 2, we define the scope of the certification schemes we are interested in, in terms of systems targeted and contributions expected. Section 3 looks at related certifications schemes and standards, arguing which parts of those do not scale to modern commercial systems and which parts do, helping to design our proposal. Section 4 investigates in further detail the requirements we state for a certification scheme in terms of agility, scalability and economic viability. We are then ready to outline the conceptual approach of the newly proposed scheme in Section 5, before investigating into the process elements included in the proposed criteria in detail in Section 6. Section 6 basically comprises a high-level catalogue of process elements that make up a best-practice secure software development lifecycle. These elements and their instances differ in terms of scope, rigor and depth, which gives rise to group them to assurance levels introduced in Section 7. In Section 8, we discuss how the process-based scheme can be extended to full Common Criteria style certification, by offering a migration path that strengthens both the process elements and their assessment. Section 9 provides an initial analysis of the proposed process-based scheme and its security claims. Finally, Section 10 concludes the deliverable.

# Chapter 2 Scope

*This chapter describes the scope of the work, in terms of the type of systems targeted (software systems and services developed in a DevOps fashion with very short release cycles and deployed on 3rd party infrastructures), the focus of the discussion (evaluation criteria for assessment of the security of a product or service, not the administrative, procedural or organisational aspects of a certification scheme) and the expected contribution to security assurance (product security properties, not security management systems).*

In order to understand and assess the contribution of this deliverable, we refine the scope of the work presented here in three different dimensions: what this document is focussing on, what we consider to be the targets of evaluation of a cybersecurity scheme based on the ideas and principles presented in the following sections, and the expected contribution of the approach to security assurance.

## 2.1 Document Focus

A certification scheme consists of two major parts: a set of criteria against which the item that is to be certified will be evaluated, and a set of processes and administrative structures that define who is entitled to perform an evaluation, how the evaluation is organised and performed, and how and under which conditions a certificate can be granted. Processes and structures can range from light-weight approaches (including self-assessment) to elaborate ones, including government authorities, e.g., for the accreditation of evaluation bodies and the issuing of certificates. The goal of establishing such processes and structures is to ensure that evaluation against the criteria is performed thoroughly and follows the same approach for each evaluation and certification effort, in the interest of having reliable and reproducible claims. To this end, a certification scheme is sometimes accompanied by an evaluation methodology, which describes in detail the actions an evaluator has to perform when assessing an item against the criteria.

In this document, we exclusively focus on the criteria part of a certification scheme. Certification aims at making a claim about the validity of certain desired properties of an item (in our case, security properties of a software artefact). To this end, criteria are defined which fulfilment give rise to some confidence in the validity of the desired properties. Evaluation and assessment activities as part of a certification effort aim at investigating if the target of an evaluation meets the criteria. Note that the certification approach does not provide guarantees for the validity of the desired properties but collects evidence that the item under investigation has passed a thorough examination against criteria which are considered to significantly contribute to the validity of the desired properties.

In some certification schemes, like the Common Criteria, the desired properties (the “security objectives”) can vary depending on the type of system or product under evaluation, the context in which it is used, and the specific risk environment. In that case, the definition of the security objectives and the related security functional requirements become part of the certification effort, with the criteria catalogue containing respective criteria for the evaluation of the requirements in terms of their adequacy and completeness.

The EU-CSA gives guidance on high-level security objectives, even though it acknowledges that for some of the products, services and processes in its scope, not all of them might apply. Article 51 states that “A European cybersecurity certification scheme shall be designed to achieve, as applicable, at least the following security objectives:

- (a) to protect stored, transmitted or otherwise processed data against accidental or unauthorized storage, processing, access or disclosure during the entire life cycle of the ICT product, ICT service or ICT process;

- (b) to protect stored, transmitted or otherwise processed data against accidental or unauthorized destruction, loss or alteration or lack of availability during the entire life cycle of the ICT product, ICT service or ICT process;
- (c) that authorized persons, programs or machines are able only to access the data, services or functions to which their access rights refer;
- (d) to identify and document known dependencies and vulnerabilities;
- (e) to record which data, services or functions have been accessed, used or otherwise processed, at what times and by whom;
- (f) to make it possible to check which data, services or functions have been accessed, used or otherwise processed, at what times and by whom;
- (g) to verify that ICT products, ICT services and ICT processes do not contain known vulnerabilities;
- (h) to restore the availability and access to data, services and functions in a timely manner in the event of a physical or technical incident;
- (i) that ICT products, ICT services and ICT processes are secure by default and by design;
- (j) that ICT products, ICT services and ICT processes are provided with up-to-date software and hardware that do not contain publicly known vulnerabilities and are provided with mechanisms for secure updates."

In Section 2.2 and 9.2, we discuss the applicability of these objectives to the targeted products and systems.

While this document focuses on the criteria to be evaluated in order to certify the cybersecurity of a target of evaluation as described in Section 2.2, it should not be read as a criteria catalogue ready to be included in a process-based cybersecurity certification scheme. It should rather be interpreted as a conceptual approach to evaluate modern commercial software products and services based on characteristics of a secure software development (and deployment) life cycle (in Chapter 5) and a non-comprehensive list of process elements that are essential for an SDLC and that can form the foundation of the evaluation criteria of a process-oriented scheme.

## 2.2 Targets of Evaluation

The EU-CSA establishes the European Cybersecurity Certification Framework, which "shall provide for a mechanism to establish European cybersecurity certification schemes and to attest that the ICT products, ICT services and ICT processes that have been evaluated in accordance with such schemes comply with specified security requirements for the purpose of protecting the availability, authenticity, integrity or confidentiality of stored or transmitted or processed data or the functions or services offered by, or accessible via, those products, services and processes throughout their life cycle." (Article 46).

In this scope, the certification approach discussed in this document target ICT products and ICT services. Even though its criteria are based on process elements for the establishment of a secure software development cycle and it is often called a process-based scheme, it should not be mixed up with a scheme targeting the cybersecurity certification of ICT processes, like, for instance, security management processes, monitoring processes, threat intelligence processes or similar. The cybersecurity certification of ICT processes is out of scope of this document. We use (development) processes and their properties as a means to increase confidence in security properties of products and services.

The products and services targeted by our approach are software products and services, i.e. software artefacts that are designed, developed and deployed by their vendors. When we talk about services, we only refer to their technical part, i.e., the functionality implemented in software that can be accessed by the consumer via a dedicated UI, an API, a mobile app (which are considered to be part of the service, since they are typically developed by the vendor of the Target of Evaluation - ToE)[1] or a browser (which is typically not part of the ToE but of the infrastructure in which the ToE is deployed and operated). Other parts that constitute a service, like service descriptions, SLAs, customer support etc, are out of scope. Given that, the distinction between products and services is insignificant for our purposes, and when we use the term software service in the remainder of this document, software products are implicitly included.

A common characteristic of the software products and services we target is that they are developed in an agile fashion, organized in development sprints, with very short release cycles, frequent updates and close integration of development, deployment and operation. Automated build processes that assemble new releases over night integrating all software that has been committed throughout the day lead to a continuous stream of new releases of systems and services ("Continuous Deployment / Continuous Integration – CD/CI") [1].

Typical examples of software services and applications we consider to be in the scope of the approach presented in this document would include:

- a mobile app accessing a SaaS-app running in a container on Amazon Web Service
- a business application integrated with

  1) an email service like Gmail or MS Outlook,

  2) analytics like SAP Analytic Cloud,

  3) an Identity provider hosted by a customer or a proprietary one, if customer doesn't have it like many startups do.

  The application is containerized and managed by Kubernetes and deployed on Amazon Web Services "for everyone", on Alibaba for Chinese customers, on Azure for public sector customers. Optionally, customers may deploy it in own datacenters (or host it by a provider of the choice).

It is on purpose that both of the above examples include cloud services, since we expect cloud services to be the largest group among the potential targets of evaluation for our approach to certification. In addition, cloud services, SaaS in particular, are expected to dominate the software business and will become one of the most important segments to be addresses by cybersecurity certification schemes.

With using the term "commercial", we also want to indicate that the products and services we target, even though they typically might touch business sensitive assets and data, are not seen to be likely to be used in high-risk environments like critical infrastructures and are not expected to be among those where only an assurance level "high" according to the EU-CSA would be acceptable. The most likely risk exposure we'd see for systems in our scope would be "moderate" to "substantial".

## 2.3 Assurance

The goal of cybersecurity certification is to provide assurance, based on evidence collected and analysed throughout an evaluation, for the security of the Target of Evaluation (ToE). Following common terminology, security refers to the confidentiality, integrity and availability (CIA) [11] of the ToE's functionality and assets, respecting defined policies for each of the aspects. For software products and services, this decomposes into two categories:

[1] We borrow the term from the Common Criteria [8].

- CIA properties of the (data) assets managed and processed by the software. These are typically addressed by introducing dedicated security mechanisms (i.e., security functional requirements and their implementation), and assurance needs to be provided with respect to their effectiveness (“are the mechanisms adequate to meet the security properties?”) and the correctness of their implementation (“do the security mechanisms function as expected?”, “are they implemented correctly?”). Objectives a), b), c), e), f) and h) of the security objectives of the EU-CSA (cf. Section 2.1) would typically be addressed by such security mechanisms.
- CIA properties of the software itself. These are typically addressed by any means that aim at avoiding vulnerabilities in the software that can be exploited by an attacker to launch an attack, for instance, by by-passing the security mechanisms of the first category. Objectives d), g), i) and j) of the security objectives of the EU-CSA (cf. Section 2.1) would typically be addressed by means in this category.

One noticeable difference between the two categories is that while the objectives and solutions for the first may vary depending on the systems business functionality, the business risk assessment, and the context, those for the second category can rather be normalized, apply to the majority of the software, and only vary in terms of general aspects, e.g., the programming language used. In consequence, to avoid unnecessary limitations of the software products and services that can be addressed by our approach, we do explicitly prescribe the security mechanisms addressing first category CIA properties, but rather require a thorough analysis of security objectives and selection of mechanisms including evidence for their adequacy.

The above approach to assurance has the additional advantage that it also applies to cases where the security mechanisms are expected to be provided by the execution environment of the service.

A notable special case occurs if the business functionality of the service is also a security functionality, for instance, an identity management service or an encryption service. Strictly speaking, the security functionality offered for consumption by the service user would be out of the scope of the evaluation and the provision of assurance, since their focus is on the security of the service rather than on its business functionality. However, this differentiation would not make sense in practice: the consumer of a security service would expect that the security certificate for such a service would also cover the offered security functionality. Hence, we would suggest to treat the security functionality offered by a service like those security mechanisms discussed in the context of the first category above, with the exception of its adequacy not being able to be judged.

## 2.4 Summary

The approach presented in this document focuses on

- Evaluation criteria for security certification of software products and services (rather than security management processes, security organisational aspects or administrative structures of a certification scheme)
- a conceptual approach to evaluate modern commercial software products and services (rather than a comprehensive and complete catalogue of criteria)
- software product and services developed and operated in an agile fashion (SaaS offers in the cloud being a typical example) and of moderate risk exposure
- a flexible approach towards security objectives (“if applicable”), security mechanisms and security functional requirements, following a risk-based approach

# Chapter 3 Related Work

*In section 3.1, we investigate in related aspects of the Common Criteria: their methodological approach to product certification, their way to derive security objectives, their most important assurance elements, their definition of assurance levels, and their (limited) support of compositional reasoning and continuous assessment. Section 3.2 looks into the ISO approach to application security. Though not being a certification scheme, ISO 27034 includes the basic elements we use for the process-centric proposal: application risk assessment, security controls selected from an organisational control library and matching the findings of the risk assessment, the definition of controls by their specification and their validation means, and more. The NIST white paper on "Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF)" provides a good summary of best practices in secure software development and is introduced in Section 3.3.*

*While the first three sections basically state the facts about the related work, Section 3.4 analyses it with respect to the feasibility for systems as described in Chapter 2.*

There are many schemes aiming at the assessment of cybersecurity, be it certification or attestation (since we are focusing on the criteria, as explained in Section 2.1, the difference between certification and attestation is not significant for our purposes). The state-of-the-art document on cybersecurity certification provided by ECSO [3] lists almost 100 schemes, designed for certifying people, organizations or products, and targeting specific sectors or product categories as well as general-purpose ones.

While cloud services are typical examples of software that meet the characteristics of systems in the scope of our approach, cloud security certification schemes like BSI C5 [47] or ANSSI SecNumCloud [48] typically address the security certification of a cloud service provider offering multiple cloud services. Hence, their focus is on the provider organization's level, including security management systems, incident management, and cross-cutting security functionalities like identity management, access control and encryption. Both C5 and SecNumCloud do include controls or requirements for Procurement, Development and Modification of Information Systems (DEV), but describe them in a generic fashion. C5, for instance, refers to policies for the development or procurement of information systems that "contain guidelines for the entire life cycle of the cloud service and are based on recognised standards and methods" (criterion DEV-01) without explicitly identifying those standards and methods, and introduces similarly generic criteria for outsourcing/procurement, development environment, testing, education, version control and deployment.

We therefore focus our investigation of the related work on those standards and methods that provide explicit requirements for the security assurance of software products and services or explicitly define best practices in secure software development or application security: The Common Criteria, ISO 27034 and, as a step towards the harmonization of the many guidelines for secure development that have been published, the NIST Secure Software Development Framework (SSDF).

## 3.1 Common Criteria

Since the mid-1990s, the Common Criteria (CC) [7], standardized as ISO 15408, serve as a reference point for security certification of ICT components and systems across sectors[2]. The CC

[2] Cryptographic modules are a notable exception, since they are typically excluded from a CC Target of Evaluation and subject to FIPS-140 assessment following NIST algorithm recommendations. Other NIST security certification schemes are, in general, sector specific.

target the security of products and systems based on the analysis of required development documentation and independent vulnerability analysis at graded levels of depth and rigor ("Evaluation Assurance Levels").

The scope of the certification is individually defined following a risk analysis and described in a so-called Security Target, which leads to a intentionally broad scope of products and systems covered by the CC. Security Targets can be schematized for given product categories (Protection Profiles). The CC are implemented as a national government scheme, with the Security Target evaluation and the product evaluation being conducted by accredited 3rd party evaluation laboratories. Mutual recognition agreements between governments exist but are limited to certain product categories and lower assurance levels. ENISA is currently working towards a proposal for a European CC-based scheme for the certification of ICT products and systems. While this will lead to a uniform approach and wider recognition of certificates across Europe, its evaluation criteria will be those of the Common Criteria, hence, our discussion of the CC will equally apply to the upcoming European scheme.

The CC target the certification of ICT products or systems (including both hardware and software) with the goal of gaining confidence in their security by understanding the security functionality (based on the reproduction of their development using the development documentation) and performing a vulnerability analysis. A Target of Evaluation (ToE) addressed by the CC is

- "a set of software, firmware and/or hardware possibly accompanied by guidance. While there are cases where a TOE consists of an IT product, this need not be the case. The TOE may be an IT product, a part of an IT product, a set of IT products, a unique technology that may never be made into a product, or a combination of these." (CC Part 1)

The operational environment of the ToE is not part of the evaluations, and it might be known (in case of a "system") or not (in case of a "product"). Since the security of the ToE might depend on characteristics of its operational environment, a security target refers to assumption on the environment. These assumptions can be validated for systems and translated into usage conditions for a product. If a customer of the product violates these assumptions when using it, the statements of the certificate do not apply.

In any case, product or system, the evaluation activities and the certificate only apply to the ToE as it has been submitted for evaluation, i.e., its particular version or instance. Any variant or update of the ToE is, in general, not covered by the certificate. The CC offer some limited support for re-certification in case of updates that do not interfere with the security objectives, the security functionality and the evaluation results, but even then require additional evaluation activities and a new certificate.

A CC evaluation covers:

- The definition of security objectives for the product/system (including the environmental assumptions)
- The definition of the security functionality to meet the security objectives
- The analysis of the effectiveness of security functionality
- The analysis of the correctness of security functionality

The CC approach is that a neutral expert performs a security assessment of the system – the evaluation – with the goal of being able to understand the rationale for the security objectives and the functioning of the security mechanisms. This understanding is gained by reproducing the specification and the development of the security functionality by the provision of evidence for the definition of the security target (or conformance to a protection profile), for the correct implementation of security functionality and for system integrity. This evidence includes documents provided by the vendor and the results of own analyses performed by the evaluator, including a vulnerability analysis.

To achieve the goal of understanding how the security functionality works, the CC refer to a variety of assurance techniques:

- analysis of development processes,
- review of development documents (architectural specification, detailed design, etc.)

- Inspection/review of source code
- analysis of the correspondence between ToE design representations,
- analysis of the ToE design representation against the requirements,
- formal security models,
- analysis of functional tests developed and the results provided,
- independent functional testing,
- vulnerability analysis,
- penetration testing
- and more

The results of applying these assurance techniques form the evidence provided for and used in the evaluation.

The assurance techniques differ in their strength, resulting in differences of the security claims that can be made using their results. The CC use this to define meaningful sets of combinations of them into Evaluation Assurance Levels (EALs) that differ in their requirements on scope, depth and rigour of the evaluation incrementally from initial to high assurance. The CC define seven EALs that are characterized as shown in Table 1.

| Evaluation Assurance Level | Characterization |
|---|---|
| EAL1 | functionally tested |
| EAL2 | structurally tested |
| EAL3 | methodically tested and checked |
| EAL4 | methodically designed, tested, and reviewed |
| EAL5 | semiformally designed and tested |
| EAL6 | semiformally verified design and tested |
| EAL7 | formally verified design and tested |

Table 1: CC Evaluation Assurance Levels

CC assurance components form the smallest entity of assurance requirements and refer to elements including developer action, content and presentation of evidence, evaluator action. Figure 1 shows an example assurance component definition for the provision and analysis of the architectural design as part of the system development.

Developer action elements:

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

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

ADV_ARC.1.3D **The developer shall provide a security architecture description of the TSF.**

Content and presentation elements:

ADV_ARC.1.1C **The security architecture description shall be at a level of detail commensurate with the description of the SFR-enforcing abstractions described in the TOE design document.**

ADV_ARC.1.2C **The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.**

ADV_ARC.1.3C **The security architecture description shall describe how the TSF initialisation process is secure.**

ADV_ARC.1.4C **The security architecture description shall demonstrate that the TSF protects itself from tampering.**

ADV_ARC.1.5C **The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.**

Evaluator action elements:

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

Figure 1: Example assurance component of the CC

Assurance components of the CC are structured into a hierarchy introducing assurance classes (high level categories referring to a specific topic area) and assurance families (subtopics within a class referring to a specific property) that allows to relate the components into a partial order of strength and to build incremental Evaluation Assurance Levels with respect to scope (which assurance families are included), depth (which level of detail is investigated in) and rigor (which level of formality is required). Figure 2 shows an example for the class ADV referring to the development of the ToE. The numbered boxes represent assurance components of increasing strength by number. For instance, ADV_FSP.4 “Complete Functional Specification” is only required for EAL 4 and higher, and there are no requirements on ADV_IMP up to EAL 3 (for software systems, this implies that the source code only needs to be provided to the evaluator from EAL 4 on).

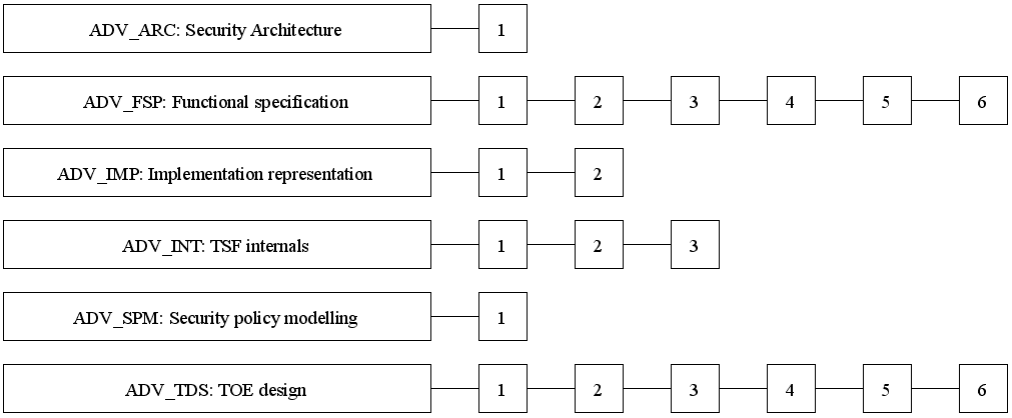


Figure 2: Example hierarchy for assurance class ADV “Development”

In their recent version, the CC include limited support for system updates and compositional certification.

As part of the product life cycle related assurance class, a family “Flaw Remediation” (ALC_FLR) is defined. It is intended to provide “assurance that the TOE will be maintained and supported in the

future" and includes requirements "for the distribution of flaw corrections". However, these are not continuously assessed, as "this family does not impose evaluation requirements beyond the current evaluation". The assurance requirements refer to the description of the procedures and the user guidance.

The composition assurance class (ACO) allows to evaluate a dependent component that relies on a base component that is certified according to the CC as well. Since some assurance components within ACO require some level of control over the base component, e.g., by requiring development documentation for the base component, the typical situation supported is that of a vendor providing both the base component and the dependent component. The base component must be a CC certified component that is not altered when used by the dependent component. Additional vulnerability analysis for the composition is required. A compositional evaluation requires access to some evaluation results of the base component (Annex B.1 of Part 3 of the CC: "Residual vulnerabilities in the base component, as reported during the base component evaluation. This is required for the ACO_VUL activities."). Figure 3 shows the relation between base component and dependent component used for compositional assurance ("TSF" abbreviates "ToE security functionality")

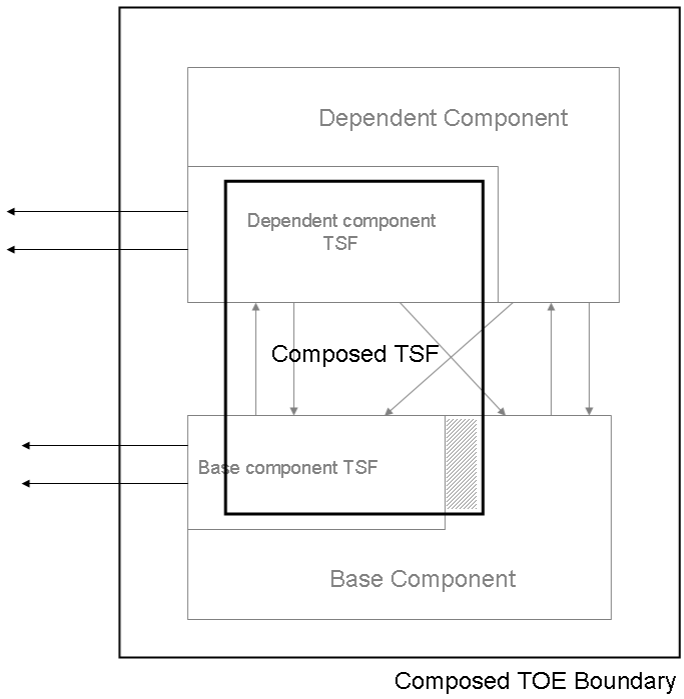


Figure 3: Composed Target of Evaluation

Like for EALs, composition assurance components are grouped in to meaningful combinations to define Composed Assurance Packages (CAPs), as show in

| Composed Assurance Package | Characterization |
|---|---|
| CAP1 | Structurally composed |
| CAP2 | Methodically composed |
| CAP3 | Methodically composed, tested and reviewed |

Table 2: CC Composed Assurance Packages

## 3.2 ISO 27034 Application Security

ISO 27034 [10] is a series of standards with the purpose to “assist organizations in integrating security seamlessly throughout the life cycle of their [software] applications by:

a) providing concepts, principles, frameworks, components and processes;
b) providing process-oriented mechanisms for establishing security requirements, assessing security risks, assigning a Targeted Level of Trust and selecting corresponding security controls and verification measures;
c) providing guidelines for establishing acceptance criteria to organizations outsourcing the development or operation of applications, and for organizations purchasing from third-party applications;
d) providing process-oriented mechanisms for determining, generating and collecting the evidence needed to demonstrate that their applications can be used securely under a defined environment;
e) supporting the general concepts specified in ISO/IEC 27001 and assisting with the satisfactory implementation of information security based on a risk management approach; and
f) providing a framework that helps to implement the security controls specified in ISO/IEC 27002 and other standards.”

ISO 27034’s approach is based on the insight that the application security risk (“the risk to an organization posed by use of a specific application”) varies with its context (business, regulatory and technological context), and that the controls applied to manage application security follow the risk assessment for the application. The management of application security for a given application therefore consists of several steps:

a) “Specifying the application requirements and environment;
b) assessing application security risks;
c) creating and maintaining the Application Normative Framework;
d) provisioning and operating the application; and
e) auditing the security of the application”

where the “Application Normative Framework” includes the definition of the context of the applications and a selection of controls (Application Security Controls, ASCs) to mitigate the identified application security risk. ASCs can be taken from a library defined for the organization, with the library being structured in terms of levels of trust, which indicate the desired security level and the controls that are needed to be applied in order to achieve that level. Given the organizational framework including the ASC library, the ASCs for a specific applications are then determined by

a) “the application's Targeted Level of Trust;
b) the organization’s requirements for the application; and
c) the application’s specific contexts and specifications”

where the targeted level of trust is an immediate result of the application security risk analysis.

Figure 4 taken from [10] shows the components of an ASC.

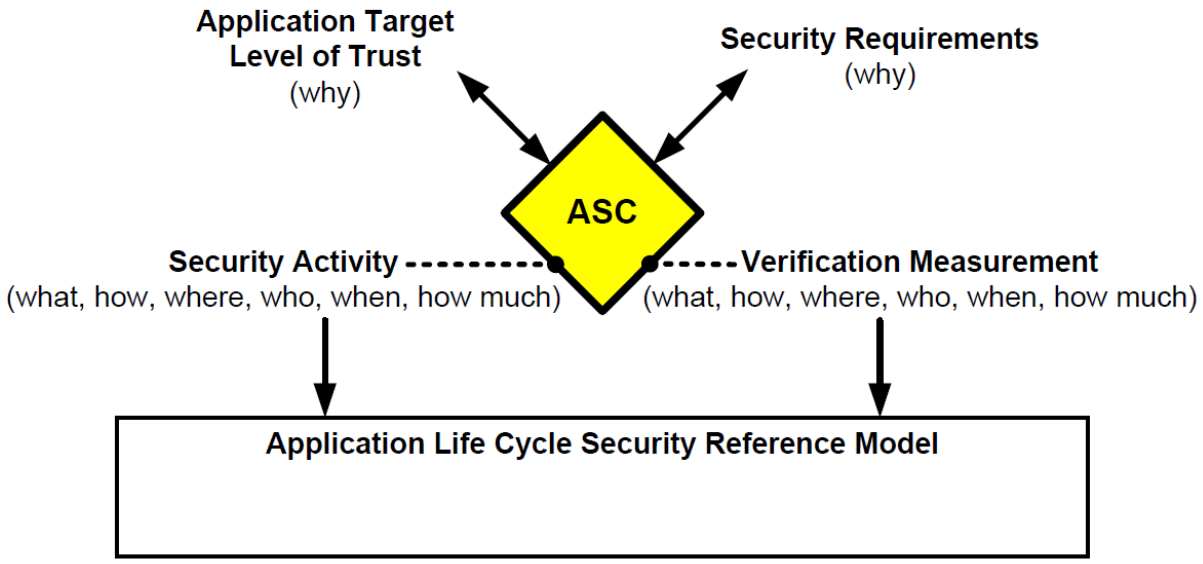


Figure 4: Components of an ASC

It is most notable that the ASC specification does not only contain a specification of the activity of the control, but also the specification of its verification measurement, i.e., “how to provide evidence that the activity was performed correctly, by a qualified actor, and that the expected results were obtained”. This is critical to ensure that control are not only described, but actually applied in a given project.

[10] states that “ASCs can be used for:

a) securing application components, including software, data, COTS and infrastructure;
b) adding security activities to processes used during stages in the application's life cycle;
c) verifying roles, responsibilities and professional qualifications of all actors involved in a project;
d) determining evaluation/acceptance criteria for components; and
e) helping to determine the application's Actual Level of Trust.”

ASC can be both technical controls (security mechanisms) or process controls (processes to be applied) and organizational controls (structures to be established).

## 3.3 NIST SSDF White Paper

In April 2020, NIST has published a white paper “Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF)” [12] which introduces and describes secure software development practices that are recommended to be integrated in a software development life cycle. The paper does not introduce new practices, but builds on a rich set of existing standards or recommendations containing or referring to such practices, and aims at structuring and unifying them into high level generic security practices that can serve as the basis for a unified approach.

The comprehensiveness of the underlying set of documents and standards allows us to refer to [12] rather than to all individual references when introducing the draft set of process elements that support a process-based certification scheme in Chapter 6. It also serves as reference for the analysis of the proposed elements with respect to their relevance and completeness.

The NIST white paper is based on the following input:

- BSIMM10: Building Security in Maturity Model (BSIMM) Version 10 [15]
- BSA: BSA, Framework for Secure Software [16]

- IDASOAR: Institute for Defense Analyses (IDA), State-of-the-Art Resources (SOAR) for Software Vulnerability Detection, Test, and Evaluation 2016 [17]
- ISO27034: International Organization for Standardization/International Electrotechnical Commission (ISO/IEC), Information technology – Security techniques – Application security – Part 1: Overview and concepts, ISO/IEC 27034-1:2011 [10]
- MSSDL: Microsoft, Security Development Lifecycle [18]
- NISTCSF: NIST, Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1 [14]
- OWASPASVS: OWASP, OWASP Application Security Verification Standard 4.0 [19]
- OWASPTEST: OWASP, OWASP Testing Guide 4.0 [20]
- PCISSLRAP: Payment Card Industry (PCI) Security Standards Council, Secure Software Lifecycle (Secure SLC) Requirements and Assessment Procedures Version 1.0 [21]
- SAMM15: OWASP, Software Assurance Maturity Model Version 1.5 [22]
- SCAGILE: Software Assurance Forum for Excellence in Code (SAFECode), Practical Security Stories and Security Tasks for Agile Development Environments [23]
- SCFPSSD: SAFECode, Fundamental Practices for Secure Software Development: Essential Elements of a Secure Development Lifecycle Program, Third Edition [24]
- SCSIC: SAFECode, Software Integrity Controls: An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain [25]
- SCTPC: SAFECode, Managing Security Risks Inherent in the Use of Third-Party Components [26]
- SCTTM: SAFECode, Tactical Threat Modeling [27]
- SP80053: Joint Task Force Transformation Initiative, Security and Privacy Controls for Federal Information Systems and Organizations, NIST Special Publication (SP) 800-53 Revision 4 [28]
- SP800160: NIST, Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems, NIST SP 800-160 Volume 1 [29]
- SP800181: NIST, National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework, NIST SP 800-181 [13]

[12] structures security practices into four groups (Prepare the organisation – PO, Protect the software – PS, Produce well-secured software – PW, Respond to vulnerabilities – RV) and contains the high-level practices as shown in

| Group | Practice |
|---|---|
| PO | Define Security Requirements for Software Development |
| | Implement Roles and Responsibilities |
| | Implement a Supporting Toolchain |
| | Define Criteria for Software Security Check |
| PS | Protect All Forms of Code from Unauthorized Access and Tampering |
| | Provide a Mechanism for Verifying Software Release Integrity |
| | Archive and Protect Each Software Release |
| PW | Design Software to Meet Security Requirements and Mitigate Security Risk |

| Group | Practice |
|---|---|
| | Review the Software Design to Verify Compliance with Security Requirements and Risk Information |
| | Verify Third-Party Software Complies with Security Requirements |
| | Reuse Existing, Well-Secured Software When Feasible Instead of Duplicating Functionality |
| | Create Source Code Adhering to Secure Coding Practices |
| | Configure the Compilation and Build Processes to Improve Executable Security |
| | Review and/or Analyze Human-Readable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements |
| | Test Executable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements |
| | Configure the Software to Have Secure Settings by Default |
| RV | Identify and Confirm Vulnerabilities on an Ongoing Basis |
| | Assess, Prioritize, and Remediate Vulnerabilities |
| | Analyze Vulnerabilities to Identify Their Root Causes |

Table 3: Security Practices according to NIST SSDF

## 3.4 Criticism and Notes

While the CC have been applied successfully in focused business domains like smart cards and firewalls, their adoption in general has been limited. In particular, this is the case for modern commercial software systems, where CC certifications only follow customer demand, typically in the Public Sector. This is mainly due to their focus on single products and systems which does not allow them to scale well to modern software development and miss the key requirements on economic viability including automation, the support of continuous assessment at manageable costs and compositionality. Each CC certificate applies to one specific version of a software and is, in general, invalidated with the next version. The certification process is lengthy and does not match the speed of product development, especially not for cloud-based software-as-a-service offers developed and deployed in a DevOps model with their extremely short release cycles. Software built on a platform with contributions from many different vendors including open source software can only be certified on a per-component base with limited value for the overall system security. While the latest CC versions include elements to address these challenges, the obstacles in practice remain: delta certifications are not supported by the criteria themselves, and the composition class is constrained by requiring access to the design information of the implemented components. To meet these challenges, the CC and the related evaluation methodology (CEM) would need to be reformed towards the support of delta certifications and taking the dependencies of systems built and operated on a (cloud) platform as well as such platforms themselves into account.

While the risk based approach supports their wide applicability, the Common Criteria emphasize the challenges of product and service certification in modern software development for the cloud by not meeting the agility needs, caused by their focus on individual product properties in contrast to the practices applied in development and operation to enforce those properties.

ISO27034 refers to application user (“application project”), i.e., the customer or user of a product or service, while the process-based approach introduced in this document explicitly addresses the vendor of a product or the provider of a service. However, since ISO27034 also contains provision for own software development, our approach can benefit from some of the conceptual ideas of ISO27034, e.g., the ASC library (which becomes the process element library in our case) or the requirements for verification measurements associated with each ASC.

The NIST SSDF white paper, published during the work on this deliverable, is an excellent summary and harmonization of the current state of the art in secure software development, by referring to a most comprehensive set of standards, methods, guidelines and recommendations for secure development practices. While we will introduce and organize a set of process elements that is geared towards the risk based approach supporting the execution of an tailored SDLC instance for each development project, the NIST SSDF will serve as a reference for evaluating the soundness and coverage of the process elements defined in Chapter 6. Section 9.3 compares the process elements with the SSDF practices in detail in order to ensure that an SDLC based on the process elements is aligned with the state of the art.

# Chapter 4 Requirements

*The scoping and the analysis of the related work lead to requirements that evaluation criteria and methodology of a certification scheme for the targeted systems need to meet: agility, scalability and economic viability.*

Certification is defined by Wikipedia [6] as the "formal attestation of the confirmation of certain characteristics of an object, person, or organization, often based on some form of external review, education, assessment, or audit". In practice, certification schemes are typically based on standards, allowing either self-declaration or 3rd party evaluation, and are driven by government or industry. Accreditation based certification is an established means to make substantiated statements about properties and functionalities of IT systems, products or services. Such statements can assist buyers in making informed decisions about the security level of software and help them to compare different solutions. While certification statements are neither able nor meant to provide any guarantee about a product's security, they increase transparency and give additional trust, based on the evidence provided and determined by the rigor of the evaluation that leads to a certificate and demonstrating that security best practices and state-of-the-art security technologies have been diligently applied. Hence, security certification plays an important role in improving cybersecurity, complementing other preventive elements of a cybersecurity strategy.

In order to maximize the uptake of cybersecurity certification, especially in the domain of modern commercial systems including cloud services, certification criteria and schemes need to be designed in a way that lead to meaningful security statements about products and services while simultaneously maintain economic viability. The latter is of particular importance in the commercial setting: conducting a certification should not introduce inadequate additional costs in terms of investments and resources needed, nor should it lead to major delays of release cycles or the go-to-market time for new products or services.

For agile and dynamic software systems, economic viability of a certification scheme means:

- Being integrated in a formal framework that guarantees that different certifiers operate on an equivalent, comparable and competitive basis and that end users can be assured that the certification is valid and comparable regardless of any specific certifying body. This includes formal approval of certification and evaluation bodies by accreditation bodies that ensure that certifiers and evaluators operate in accordance with standard common methodologies to comparable levels of rigor and scrutiny.
- Being of global scale, spanning nations, verticals and organizations. If schemes are instantiated under the auspices of individual nations or organizations, mutual recognition agreements, like they exist, for instance, to a limited extent for the Common Criteria, are essential. Otherwise, the need for multiple efforts – either to support different schemes or to repeat certifications under different (national) regimes for the same scheme – would increase the costs of certification for globally operating vendors to a level that will not be rewarded by the market. Such recognition agreements should be strictly implemented in order to avoid their circumvention and to provide planning reliability to the vendors.
- Having a scope covering a variety of products, services and contexts, so that organizations can establish routines to support certification across their product portfolio. This means that the security objectives and functional requirements for a target of evaluation cannot be uniformly stated and should be following a risk assessment taking its specific context into account.
- Allowing for continuous assessments matching the release cycles of modern commercial software systems. Given the criticality of fast release cycles (weekly, daily or even hourly) and the automated build technologies supporting such a tight schedule, continuous assessment including the maintenance of the certificate needs to show a high degree of automation as well. Dynamic changes of the behavior of systems, e.g., caused by using higher-order programming concepts like Java reflection, need to be included in the assessment.
- Supporting the usage of claims or certificates about components of the system or infrastructure the system is relying on, as well as the validation of assumptions on those components. The support of compositional reasoning about security is essential when systems resulting from complex supply chains should be in the scope of a security. certification.

In the following, we look in closer detail into these requirements by translating them into five requirements that our proposal criteria will be analysed against. However, we do not further discuss the first two items list above, since they refer the organizational and administrative framework of the certification scheme. According to Section 2.1, this document focuses on the assurance and evaluation criteria, hence, those requirements are out of scope of the work presented here.

## 4.1 Security Characteristics

Following the definition of certification [6] and the scoping in Chapter 2, we aim at criteria for the attestation of the confirmation of security characteristics of software products and services. The desired security characteristics include:

- Protection against vulnerabilities in the software, including during its operation
- Protection of assets managed, controlled or created by the software, expressed in terms of maintaining their CIA properties and achieved through the design and implementation of security mechanisms
- The maintenance of the protection during the lifecycle of the software

This generalized definition comprises the security objectives stated in Article 51 of the EU-CSA (cf. Section 2.1)

The goal is not to provide guarantees for the validity of these desired characteristics, but to increase the confidence, because the evidence produced and analyzed during the evaluation give rise to assuming their validity.

**R0: A certification scheme for modern commercial software products and services should aim at increasing the confidence in the validity of the desired security characteristics.**

## 4.2 Complexity

Today's commercial software systems are characterized by a high level of complexity. This complexity is caused, among others, by:

- Rich dependencies: software products and services may include large open source packages, even if only a part of its functionality is used. Studies at SAP have shown that typical software contains both direct (~20%) and indirect dependencies (~80%) on open source software (OSS) components, with ~95 dependencies per module, not including build dependencies
- Size: due to the tight integration in software development and execution frameworks, the use of large OSS packages (where typically only a portion of the functionality and code is used), virtualization/containerization, and others, the size of a modern commercial software application or component tends to be big, including hundreds of thousands of lines of code.
- Limited control: SAP internal studies [30] have shown that, while 20 years ago 95% of the software of a typical SAP product was developed in-house, this ratio has now been reversed. Foreign code, including browser, language engines, microservices, application servers, container operating systems, virtualization (Kubernetes, Docker, CloudFoundry, etc.) and operating systems make up 95 % of the code base of a modern commercial application. This foreign code is not contracted from a vendor, but comes "from the internet", meaning that there is no or only limited control over its development and the processes applied to it.
- Delegation of security functionality: This foreign code also includes security functionality like secure communication protocols or identity providers. And while it is good security practice to use established implementations of security functions rather than developing own ones, it also means that some types of assurance evidence required for product certification, like detailed development documentation, might not be accessible and need to be replaced by alternative types of assurance.

An example application illustrating these complexities has been introduced in Section 2.2 and consists of

- a business application integrated with

  1) an email service like Gmail or MS Outlook,

  2) analytics like SAP Analytic Cloud,

  3) an Identity provider hosted by a customer or a proprietary one, if customer doesn't have it like many startups do.

  The application is containerized and managed by Kubernetes and deployed on Amazon Web Services "for everyone", on Alibaba for Chinese customers, on Azure for public sector customers. Optionally, customers may deploy it in own datacenters (or host it by a provider of the choice).

The proposed certification scheme should be able to cope with all of the dimensions of complexity indicated above. A software product or service should not be prevented from being certified – with respect to the evaluation criteria defined – because it relies on and is part of a software ecosystem where multiple entities provide infrastructure and code components that are shared across the ecosystem.

**R1: A certification scheme for modern commercial software products and services must allow to certify complex systems with rich dependencies, large size and limited control of the ToE owner/vendor. It has to take into account that essential security functionality (for instance, Identity and access management) is outsourced, i.e., not part of the ToE, and provided by a 3rd party (for instance, the cloud infrastructure provider).**

## 4.3 Risk-based Approach

The proposed scheme is meant to cover software products and services of all kind: low sensitivity or high sensitivity[3], on-premise or deployed in the cloud, applications or infrastructure, mobile or web applications, consumer or business software, and more. The risk exposure of such systems as well as their required security level differs, and so do the requirements for assurance that give the buyer or the consumer an adequate level of confidence in the security of the software.

To keep the scope broad and to manage the certification effort at an adequate level, we follow the idea of the Common Criteria and of ISO 27034 and do not prescribe particular security objectives, security requirements (in our case, process elements) and assurance elements for every evaluation, but allow to come up with a meaningful selection of elements for the individual ToE, provided their adequacy for the demonstration of the security of the ToE is shown.

For security objectives and security requirements this means that a risk analysis is performed on the ToE and its environment, leading to the definition of security objectives where process elements are then selected (e.g., from a given library) that support the achievement of the objectives. This selection of process elements compares to the security functional requirements stated in the security target; however, they are not strictly functional requirements since they refer to process steps in the development or operation of the ToE.

For assurance elements, this means that meaningful combinations of elements can be combined into assurance levels that reflect a certain level of scope, depth and rigor adequate for the evaluation task at hand.

**R2: A certification scheme for modern commercial software products and services should support a risk-based approach to certification, with the elements for protection and**

[3] Even though we have focused on medium-sensitive systems in the scope definition in Section 2.2, the scheme itself should not prevent including systems of lower or higher sensitivity.

**assurance (process elements, evidence required and vendor and evaluator actions defined) being selected based on a risk assessment for the individual ToE. To allow comparisons, sets of elements considered to be adequate for typical risk postures can be combined into assurance levels.**

## 4.4 Scalability

### *4.4.1 Full Life-cycle Support*

A frequently stated concern about current approaches to cybersecurity certification, the Common Criteria in particular, is that the certificate is issued at a certain time with its claims only applying to the ToE as it is presented at that point in time. Systems targeted by cybersecurity certification, though, are operating in a highly dynamic environment, with changing risk exposure, new threats becoming known, and new technologies available to attackers or deployed in the system's operational environment introducing potential new vulnerabilities. Hence, evaluation criteria for modern commercial software should also cover life cycle phases beyond design, development and deployment and include system operation.

A common approach is to cover life cycle aspects in the administrative part of the scheme, for instance, in the auditing approach. This includes, for instance, the limitation of the validity period of a certificate, spot checks or tests of the ToE after issuing the certificate, and the requirement to provide evidence for operational effectiveness. Since our proposed approach to certification is based on process element applied to the ToE, we can also include life cycle related process elements in the criteria for certification, indicating that processes covering the later life cycle phases are in place and are applied throughout the life cycle.

The CC include the assurance family ALC_FLR "Flaw Remediation". It is intended to provide "assurance that the TOE will be maintained and supported in the future" and includes requirements "for the distribution of flaw corrections". However, these are not continuously assessed, but the ALC_FLR requirements refer to the description of the procedures and the user guidance. In our proposed certification approach, we also want to include requirements that ensure that these procedures are applied throughout the life cycle

**R3: A certification scheme for modern commercial software products and services should support the full life-cycle of a software product or service, including its operation. Assurance elements relating to the maintenance of the security of the product or service, including updates and security patches, and its continuous assessment, should be included in the criteria.**

### *4.4.2 Composition*

Software has always been built in a modular fashion to manage its complexity, but with the increasing number of open source software projects and the ease of consumption and accessibility of the code resulting from OSS projects, this trend has been amplified. As a result, modern commercial software is highly compositional, with components stemming from a wide variety of 3rd party contributors. Components include large pieces of software (like a web application server or a containerization framework) as well as small ones (like microservices). In many cases, a software product or service consists mainly of 3rd party contributions, with the product or service vendor only adding some glue code that ties the components together.

The security quality of the components varies significantly, and it would be unrealistic to assume that commercial software builds on security certified components only. Our proposed security evaluation criteria should therefore include process and assurance requirements that allow to draw conclusions about the security level of a component. The more is known about the security of the components (e.g., evidence for assurance being available, results of approval procedures, security properties being part of an SLA, components being certified), the fewer and less rigorous requirements will be

imposed on component-related assurance activities of the vendor or evaluator of the software under evaluation. When defining such actions, it needs to be considered that no further information about the component might be available and it has often to be treated as a block-box.

It is a well-known fact that security is non-compositional [31], i.e., composing two locally secure components does not, in general, lead to a secure system, and

$$\frac{S_1 \vdash \varphi_1,\ S_2 \vdash \varphi_2}{S_1 \parallel S_2 \ \vdash\ \varphi_1 \bigwedge \varphi_2}$$

does not hold. One way to address the compositionality problem is to restrict the characteristics of security properties and system specifications in a way that allows for compositional reasoning. That comes at the expense of possibly not being able to reason about all properties of interest but allows to efficiently and effectively analyze those that meet the restrictions. If one can find restrictions such that still a large set of properties of interest remains included, high practical relevance is given. And if such restrictions are not met by a system, this is likely to give hints for potential vulnerabilities and attack vectors.

[31] introduces some general principles that allow for compositional reasoning while allowing a large set of properties and systems to be included. First, they focus on safety properties (i.e., properties of the type “nothing bad happens”, in contrast to liveness properties of the type “something good will eventually happen”), which allows to consider finite traces of systems only (“trace semantics”). Second, they demonstrate that security reasoning is compositional if the required properties can be expressed by local properties of trusted entities (e.g., “if entity S sends message B it has received message A before”) and interface invariants of the system (e.g., “an adversary cannot change the signature of interfaces”). Combined with rely/guarantee-style proofs, the latter applies to inductive security properties as well. And while the restrictions appear to be significant, the authors demonstrate that many relevant security properties and systems meet them. They have applied compositional reasoning to the security analysis of security protocols, trusted computing, file systems, operating system, and more.

A certification scheme that supports compositional reasoning must require a compositionality analysis based on the above considerations to demonstrate that compositional reasoning is valid for the case at hand.

**R4: A certification scheme for modern commercial software products and services should support the certification of composed systems using components, services and infrastructures from foreign entities or OSS, including those based on components that are not certified themselves. In case of using non-certified components, the scheme must provide assurance elements justifying that those components do not impact the security of the ToE, or that, in case of components providing security functionality, there is evidence that the provided functions are functionally effective and correctly implemented.**

## 4.5 Agility

Agile development methods like Scrum [32] are now the dominant paradigm for software development. Their incremental and fast release cycles dramatically reduce the go-to-market time, and they allow for fast releases of a first viable product as well as for adding new features at unprecedented speed. Whereas this is incredibly advantageous from the business point of view, it imposes challenges for certification by independent evaluators and based on evidence collection for assurance: performing the assessment for certification targeting only one version during this continuous release cycle cannot keep pace with the speed at which new releases are produced.

Since our intent is to increase the level of security for commercial software products and services through certification, any attempt to change these market dynamics in support of security certification is like to fail this intent. What we need instead are certification schemes based on criteria that can cope with the agility of modern software development. In Chapter 5 and Chapter 9 we will argue that

this can be achieved by focusing on the invariants of agile development: the methodology and the security related process elements that are integrated in the methodology, rather than by focusing on the outcomes of these process elements.

**R5: A certification scheme for modern commercial software products and services should provide the necessary agility to not impact the release frequency and speed of modern commercial software systems including releases stemming from fully automated build processes. The criteria should allow for the certification of a continuous series of releases that are provided over time with a minimum of additional effort per release (ideally, in an automated fashion).**

## 4.6 Manageable Efforts

A prohibitive element that limits the practical application of security certification schemes like the Common Criteria are the costs and efforts that are caused by certification. Figure 5 taken from [33] suggests that the certification costs increase linearly with number of products certified (while the costs for the site certification are a one-time effort).

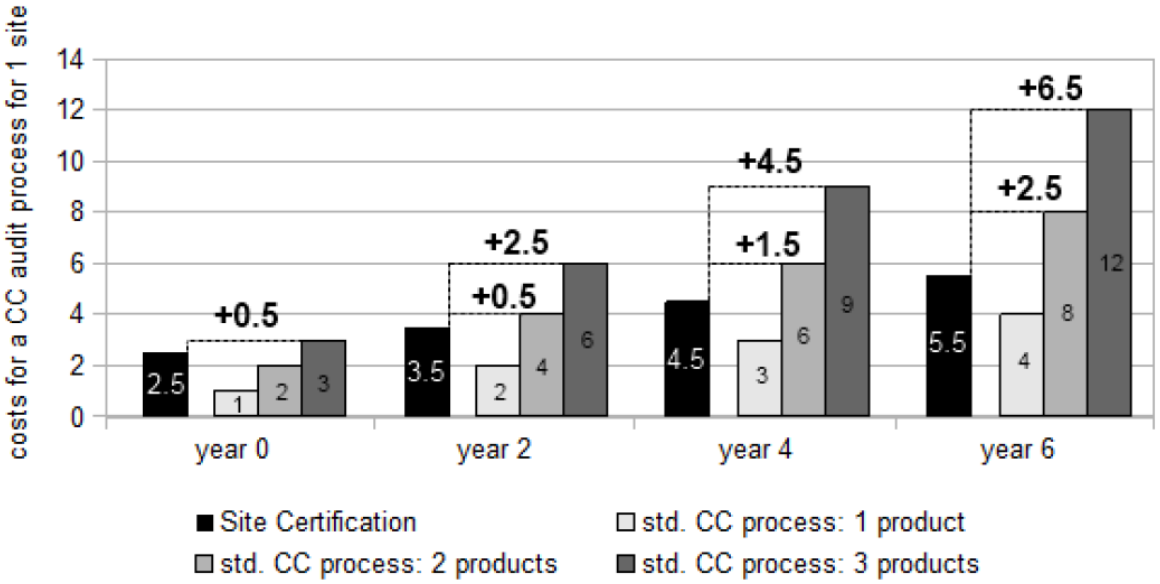


Figure 5: CC cost schematics

The costs and efforts of a CC certification may vary significantly, but as a rule of thumb, [34] states that “It is safe to assume that the cost will be north of USD $100k.” and that with respect to the duration of a CC certification “The general rule of thumb is about one year including preparation time using the legacy approach.”, even though automation solutions provided by the authors of [34] can lead to “evaluations … be completed in 2-3 months from end to end.”

The indications above allow to conclude that such figures exceed reasonable values acceptable for commercial software development in an agile fashion with ultra-short release cycles, and that in order for a cybersecurity certification scheme to be accepted in the commercial market, its costs need to be manageable and drastically reduced compared to the CC approach.

**R6: The costs and efforts required for the certification of modern commercial software products and services should be limited to an amount that can be rewarded by the market.**

# Chapter 5 Outline of the Certification Approach

*The general principles of a process-centric approach to security certification are introduced. It is explained why a focus on development processes and environments is more likely to meet the requirements of Chapter 4, which processes are of particular relevance and how related criteria can be designed. The impact on evaluation activities and methodologies is discussed.*

## 5.1 General Approach

In order to reflect the current paradigms in the software industry, we argue that a cyber security certification scheme should, in general, be process-based, with certification evaluating the outcomes of the processes for the individual product or system only required for high risk environments like critical infrastructures with a long lifetime for the installed technology components. In the following, we discuss some methodological concepts and elements of evaluation criteria of a process-based approach, which would help to meet the requirements stated above.

For the software and (cloud based) services industry, we promote a certification scheme that focuses on the effectiveness of the processes that are applied to the development, deployment and operation of secure software and that is based on international standards. Effectiveness of a process includes process definition, enforcement of the process application, automation of the process, and means for validating the process application. With process certification, one can provide the required insights in the security best practices applied in the development activities that each product undergoes, acknowledge different protection needs and risk exposure, as well as scale to fast release cycles and cloud operation models. It is possible to extend the scope of certification to all lifecycle phases of a software product, including deployment and operation (with elements such as, for instance, regular updates or patch management). Process oriented schemes are also better prepared for technology evolution, both for business functionality and security functionality, in that they do not require complete re-evaluation when new product versions embody the latest technology, provided these are controlled by the best-practice methodologies that were the target of the process certification.

A process-oriented security certification scheme for software products and services targets product, system and service security by investigating into how they are developed in an organization. It focuses around the establishment of a secure development life cycle (SDLC) [8] and secure operations of cloud services. Based on the assumption that well-defined and rigorously applied process elements that match the state-of-the-art in security-by-design, security testing, secure operations and more lead to a predictable security quality of the outcomes of such processes, processes that are certified once lead to security claims about many products provided there is evidence that the processes are indeed applied.

## 5.2 Process Elements

A secure development life cycle consists of a number of activities related to the specification, development, testing, deployment and operation of a secure software product or service[4]. We call those activities process elements, and the combined application of these process elements constitutes the (security part of the) SDLC or an instance of the SDLC for a given product or service.

[4] Note that in the context of this document we exclusively focus on security related activities and leave other development life cycle activities out of scope.

Process elements are comparable to practices of [12] in that they describe general and abstract security activities at different phases of the software development and operation.

Process elements may vary in scope, depth and rigor of their included activities and the documents or evidences that come out of their application (for instance, specifications, test protocols, code or proofs).

- Scope: The parts of the software under evaluation (the ToE) that are included in activities required by the process element (e.g., the security modules or the whole software)
- Depth: The level of detail on which the activities required by the process element are performed (e.g., a specification of the source code)
- Rigor: The strength of the methods applied by the activities required by the process element (e.g., a document review or a formal proof)

Process elements themselves can occur in different instances distinguished by scope, depth and rigor. For instance, security testing can be done by spot checks, or be based on a thorough coverage analysis.

An organization implementing an SDLC is expected to have a number of security related process elements in place – a security process element library in analogy to ISO 27034's ASC library –, and to apply those process elements to each of their development efforts resulting in a software product and service that is secure by design. Additional elements applied to the deployment and operation of the software product or service aim at maintaining the security properties achieved during development and provide updates if vulnerabilities are discovered or the threat landscape changes.

The risk associated with different products or services developed and offered by an organisation may vary, depending on business functionality, technology used, dependencies on 3rd party providers and OSS (including indirect dependencies), operational environment or other factors. Depending on the outcome of the analysis of that risk, the organisation may decide to use different combinations of SDLC process elements or instances of process elements for each of their development projects. Our approach for a process-based scheme for security certification for software products and services supports the risk based approach to the selection of process elements by including a risk analysis element (comparable to the application risk analysis of ISO 27034 [10]) and a security planning element that describes the selection and its justification based on the results of the risk analysis. These two elements, risk analysis and security planning, are mandatory elements and cannot be left out when defining the specific instance of the SDLC for a given product or service.

If the security of a product or service is assessed based on criteria referring to the processes applied during their development and operation, it is important to ensure that documented processes are indeed applied. We therefore adopt the concept of ISO 27034 and require process elements to consist of both their documentation as well as validation measurements for their application. Such validation measurements can refer to logs indicating the application of a process element, to checking the existence of the expected outcomes of a process element (e.g., a specification, a test protocol or a report of a static code analyser) or to signed statements by the vendor. The validation depends on the respective process element and will be detailed per element in Chapter 6.

In general, our approach to risk analysis and control selection is inspired by and comparable to ISO 27034. Notable differences occur with respect to the type of controls included (process elements of an SDLC only) and the risk assessment. Since we address the software vendor, the business, regulatory and technology context might not be known in advance and need to be assumed with the assumptions made explicit in the risk assessment. The technical context is only available for those cloud services where the infrastructure on which the service is deployed is known. If in addition the service is deployed on infrastructures owned by the provider (parts of), also the business and regulatory context might be known.

## 5.3 Definition and Application of Process Elements

Process elements are at the core of a process-based security certification scheme for products and services: the security assurance provided by the scheme is based on the application of a tailored (with respect to the risk assessment) SDLC, which itself is referring to their definition, their selection, their application, and the validation of their application.

In Chapter 6, we describe process elements according to the template in Table 4.

| Item | Description |
|---|---|
| ID | Process element identifier |
| Name | Process element name |
| Purpose | Security objectives related to the application of the process element |
| Description | Description of the activities performed during the application of the process element |
| Output | Description of the expected outcome of the application of the process element |
| Validation | Description of the activities performed and measurements taken in order to demonstrate that the process element has been applied and produced the desired outcome |
| Elements | Describes variants or instantiations of the process element that differ in scope, depth and rigor, in order to adapt the activities and the methodologies used to conduct them to the risk assessment |
| References/dependencies | Describes relations between process elements, in particular dependencies, i.e., if selection of the process element requires the selection of other process elements or excludes them from the selection |
| Mandatory | “yes”, if the inclusion of the process element in the selection is required in any case, “no” otherwise |
| Application Notes | Additional comments and guidelines for the selection or application of the process element |

Table 4: Process element description template

Application of process elements for a product or service means that the activities described for the process element are executed and the related outputs are produced during development, deployment and operation of the product or service.

## 5.4 Validating the Application of Process Elements

A process-based security certification scheme for software products and services can only be effective if the processes are not only described and the description assessed to find the process elements and their combination being adequate to result in secure software, but if there is also

evidence for their application in the development, deployment and operation of the products and services. In our case, “application of a process element” means that the activities referred to in the element description are executed and the required outcomes have been produced.

For instance, if the process element describes security testing, activities may include the specification of test cases, the execution of the tests and the recording of the test results in a test report, but would also require a verdict of the positive outcome of the tests, i.e., the recorded test results match the expected results. The process element would not refer to the individual test case descriptions (they are typically product and service dependent), but require that none of the specified tests should fail.

The evidence for process application can be produced in a number of ways, impacting the strength of the application claim and the effort required for its assessment.

- For each product or service, the existence of documents as required by the “output” part of the process description (P1)
- For each product or service, the existence of documents as required by the “output” part of the process description with the document content checked for plausibility (P2)
- For each product or service, the existence of documents as required by the “output” part of the process description with the document content evaluated and analysed for quality and completeness (P3)
- A random spot check across different products or service by the same vendor, or across the different process elements within the development, deployment or operation of one product or service, for each of the above dimensions (R1, R2, R3)
- An enforcement process that is established by an organization and evaluated to ensure the application of process elements and the validity of the requirements on their output, for each dimension of the first three items in this list. (E1, E2, E3)

These different ways for producing evidence of process element application form a hierarchy (more precisely, a lattice) with respect to the strength of the application claim:

1) P1 < P2 < P3
2) R1 < R2 < R3
3) E1 < E2 < E3
4) R1 < P1
5) P1 < E1
6) R2 < P2
7) P2 < E2
8) R3 < P3
9) P3 < E3
10) Least upper bound: E3
11) Greatest lower bound: R1

The ordering relation of the lattice will support the definition of different assurance levels for a process-oriented certification scheme.

## 5.5 Assessment Methodology Principles

How can process elements be used to provide assurance for the security of a software product and service? In this section, we describe how an assessment aiming at the provision of assurance can be performed. The general idea as described in Section 5.1 is that of combining process elements based on a risk analysis into a reasonable instance of the SDLC for the product or service under evaluation and to monitor the successful application of these process elements. The assessment therefore comprises of

1) An evaluation that the risks relating to the product or service in the assumed or actual context have been properly identified

2) An evaluation that the instance of the SDLC chosen for the development, deployment and operation of the product or service (i.e., the process elements selected) is adequate with respect to the identified risk
3) An evaluation that all of the selected process elements have been successfully applied or will be applied during the operation of the product or service after the time of the evaluation

If all these evaluations have been performed and led to a positive verdict, the resulting overall security claim is

**"The product or service is developed, deployed and operated by applying a state-of-the-art SDLC matching the product/service risk, without noticeable problems or conspicuous features being detected."**

In order to perform an assessment, evaluation tasks on both organisational level and product/service level are defined. Organisational level tasks scale across multiple evaluations of products and services and need to be performed only once unless changes in the related parts of the organization occur. If several products or services share a common risk assessment, the resulting SDLC instance can be shared as well.

### *5.5.1 Combination of Process Elements*

To keep the scope broad and the efforts adequate, the proposed process-based approach to security certification does not refer to a single, uniform definition of the SDLC, but allows to tailor it according to a product/service risk analysis. Performing the risk analysis and instantiating the SDLC are process elements by themselves, and are included in the set of process elements. However, their inclusion in the definition of the individual SDLC instance must be enforced for the working of the approach, hence, they are considered as mandatory elements which cannot be deselected.

The individual SDLC instance is a combination of process elements as they are defined in Chapter 6 of this document. As the process elements are defined in a pretty general way, their actual occurrence in an SDLC instance will be more detailed, referring to the variants indicated in the descriptions, or even in further detail. For each occurrence of a process element, it has to be demonstrated that they are indeed a specialization of the respective process element, together with an indication of its strength.

While in general the approach does not require that an organization normalizes its process element occurrences, the introduction of a process element library in analogy to ISO 27034's ASC library is advantageous. It provides a reference for the selection of elements for a given assessment, and the demonstration of their consistency with the element specification in the evaluation criteria and of their strength can be done as a one-time effort on organisational level.

The combination of process element occurrences into an SDLC instance for a given evaluation of a product or service has to be assessed with respect to their consistency (which is implied if the selection is taken from a process element library) and completeness. By completeness we mean that potential dependencies between the process elements are respected and that the strength (i.e., scope, depth and rigor) of the selected process element occurrences is adequate with respect to the risk analysis performed. If assurance levels are defined for the scheme and addressed for the actual evaluation, the requirement for adequacy extends to the assurance level targeted.

### *5.5.2 Organizational Level*

A number of assessment activities required for awarding a certificate with respect to the scheme can be performed on the organisational level, and the assessment result apply to all product or service developments performed in this organisation. This is one of the keys for achieving scalability and agility: Assessment tasks performed on organisational level can be performed as one-off tasks, and do not slow down the release time for individual products and services including their updates.

Assessment tasks on organizational level include:

- Checking the quality of the process elements defined for the organisation:
  Analyze the process element library defined for the organisation for adequacy with respect to the process element definition of the criteria, for consistency and for completeness. In case of various process element occurrences of the same process element in the library, analyse the occurrences for their relative strength.
- Checking the approach for selecting the process elements:
  Evaluate the process element “SDLC instance definition” with respect to the activities defined for this process element for the organization, in particular, the use oif and reference to the results of the “Product/service Risk Assessment” process element
- Making spot checks for the results of the selection of process elements:
  Analyse selected previous product or service development projects of the organization for the application of the process element “SDLC instance definition” and its results.
- Evaluate enforcement mechanisms for application of process elements
- Making spot checks for the application of process elements
- Analyse selected previous product or service development projects of the organization for the application of (selected) process elements and their results.

The purpose of including assessments of previous projects of the organization is to get insights into the discipline of applying the SDLC at the organization, which gives hints for its application in the actual effort.

### *5.5.3 Product/Service Level*

The intent of the scheme is to accomplish an economically viable certification. The approach to achieve this is to minimise the assessment tasks at product/service level and to keep their effort and costs low. If the organizational level assessment tasks are performed as indicated in Section 5.5.2, only the following tasks need to be performed on product/service level:

- Check that the risk analysis has been performed:
  Depending on the assurance level targeted, this can be limited to the determination that the risk assessment has been done up to a detailed evaluation and assessment of its results;
- Check the selection of process elements:
  Depending on the assurance level targeted, this can be limited to the determination that the process element selection has been done following the processes defined for the organization up to a detailed analysis of the adequacy of the selection with respect to the risk assessment results
  We aim at making only a minimum number of process elements mandatory, theoretically allowing pathological cases like not selecting any element. This evaluation activity is designed to detect and avoid such cases.
- Check the evidence for process element application
  For each process element selected, its application should be validated according to the validation requirements in the process element description. In case of the organization having defined enforcement processes for the application, this task is part of the organization level assessment.

## 5.6 Towards a process-oriented Software Security Certification Scheme

Even though the definition of a full certification scheme based on the approach described above is out of scope of this document, we can already indicate some of the characteristics that such a scheme would show.

- The evaluation comprises of two parts: an organizational level evaluation and a process/service specific evaluation.

- The organizational level evaluation is performed once and applies to the entirety of products/services developed and delivered through this organisation, while the process/service specific evaluation has to be performed for each product/service developed and delivered through this organisation individually.
- The differentiation of process element occurrences and evaluation activities with respect to their strength supports the definition of assurance levels (cf. Chapter 7)
- The security requirements against which the evaluation is performed are defined by the process elements introduced in Chapter 6. This is implemented by the process elements becoming part of the evaluation criteria.
- The criteria do not prescribe the requirements out of the set of process elements that need to be met for an individual process/service evaluation. This depends on the risk analysis results, the assurance level targeted (if applicable) and the following process element selection.
- In order to facilitate this flexible approach without compromising the security claims of the certificate, the process elements (or security requirements) for risk assessment and process element selection are mandatory.

Note that we do not include considerations regarding the CAB (Conformance Assessment Body), accreditation procedures, the processes relating to the actual issuance of the certificate, the question of self-declaration of conformance, the documents published with a certificate, and other administrative or regulatory aspects here, since they are out of the scope defined in Chapter 2.

# Chapter 6 Process Elements

*This chapter details the process elements, and is meant as the starting point for a catalogue of process elements that can become the core of a process-centric security certification scheme. Process elements are characterised by their contribution to the system security and the means used to evaluate their quality and application to a given development effort. Process elements that target the same security quality can be put in a hierarchy according to their scope, rigour and depth*

In this chapter, we introduce a set of process elements and their description that can form the basis for further refinement towards a process element catalogue becoming part of the criteria for a process-based software product and service certification scheme.

The set of process elements consists of:

PE1 Organizational Security Framework

PE2 Product/Service Risk Assessment

PE3 SDLC Instance Definition

PE4 Security Planning

PE5 Software Architecture

PE6 Threat Modelling

PE7 Security Functional Requirements Definition

PE8 Secure Programming Guidelines

PE9 Code-level Security Analysis

PE10 Security Testing

PE11 Security Assessment of 3rd Party / Open Source Software

PE12 Assessment of the Operational Environment

PE13 Development Environment

PE14 Vulnerability Analysis

PE15 Continuous Vulnerability Checks

PE16 Patch / Update Processes

PE17 Secure Configuration by Default

PE18 Secure Deployment

PE19 Formal Modelling and Analysis

PE20 Tools and Automation

PE21 User Guidance

We introduce each process element in tabular form following the template defined in Section 5.3, Table 4: Process element description templateTable 4.

## 6.1 Organizational Security Framework

| Organizational Security Framework | |
|---|---|
| ID | PE1 |
| Name | Organizational Security Framework |
| Purpose | Provide the environment within an organization to define and execute SDLC instances for product or service projects |
| Description | The Organizational Security Framework defines roles & responsibilities, establishes and documents the process element library, defines security levels for the organization (i.e., SDLC instances for groups of products and services developed and operated by the organization with a common risk level). If the organization has procedures for the enforcement of process elements application in place, they are part of the Organizational Security Framework.<br>Based on the process element library, the Organizational security Framework can define organizational assurance levels, i.e., standard combinations of process elements for the organization reflecting the security sensitivity of products or services. |
| Output | Process element library, policies, documentation of the Organizational Security Framework, enforcement procedures and related processes and tools. |
| Validation | Documents, procedures and tools are in place and meet requirements on content and presentation |
| Occurrences | Enforcement procedures can range from documents & descriptions over tool-based workflows to quality gates with approval procedures for the individual process elements. |
| References/dependencies | None |
| Mandatory | No |
| Application Notes | We do not formally require an Organizational Security Framework to be in place. However, the process-based certification will yield more significant advantages when an Organizational Security Framework is set forth, and multiple certifications are run under its control, for instance, for each new release of a product or service. |

Table 5: Organizational Security Framework

## 6.2 Product/Service Risk Assessment

| Product/Service Risk Assessment | |
|---|---|
| ID | PE2 |
| Name | Product/Service Risk Assessment |
| Purpose | Perform a project specific risk analysis and assessment for a product or service development / deployment / operation |
| Description | Product/Service Risk Assessment performs an analysis of the security sensitivity of the given product or service by taking the business, regulatory and technology context into account. It aims at defining at justifying the security level that is required for the product and service, as well as the related assurance level. |
| Output | Document the major risks for the individual project, identify the required assurance level |
| Validation | Risks documented on paper or in a tool, risks being clearly identified so that they can be traced throughout the project, analysis of risk analysis methodology applied and justifications provided for the individual risks. |
| Occurrences | From paper documentation over tool-based risk management to semi-formal or formal specifications. |
| References/dependencies | PE1: for the definition of organizational assurance levels, i.e., standard combinations of process elements for the organisation reflecting the security sensitivity of products or services. |
| Mandatory | Yes |
| Application Notes | The Product/Service Risk Assessment aims at identifying the security sensitivity of the product or service. It therefore focuses on the risks associated with the managed assets and data as well as the context. It does not include a detailed technical risk assessment (e.g., which technical component of the product is part of the attack surface), this will be addressed in PE6 “Threat Modelling”.<br>The major difference can be characterized by PE1 aiming at the instantiation of the SDLC, while PE6 aims at the identification of security functional requirements and security mechanisms needed. |

Table 6: Product/Service Risk Assessment

## 6.3 SDLC Instance Definition

| SDLC Instance Definition | |
|---|---|
| ID | PE3 |
| Name | SDLC Instance Definition |
| Purpose | Define the individual SDLC instance (i.e., combination of process elements) for the product or service |
| Description | The SDLC instance definition defines the process elements from the process element library that must be applied to the given product / service development / deployment / operation. The SDLC instance definition is based on the result of the Product/Service Risk assessment (PE2) and includes only and all the elements required for the intended security and assurance level. |
| Output | Specification of the SDLC instance of the given product or service |
| Validation | Documentation of the process element selection, instantiation of procedures and tools required by the selected elements |
| Occurrences | From paper-based documentation to tool-based enforcement |
| References/dependencies | PE1: The Organizational Security Framework defined the process element library from which the elements of the SDLC instance are taken, it also provides the procedures and tools to execute the SDLC instance and provides the related outputs and validation documentation.<br>PE2: The SDLC instance is required to reflect the results of the Product/Service Risk assessment |
| Mandatory | Yes |
| Application Notes | -- |

Table 7: SDLC Instance Definition

## 6.4 Security Planning

| Security Planning | |
|---|---|
| ID | PE4 |
| Name | Security Planning |
| Purpose | Plan security activities including the application of the process elements of the SDLC instance for the product/service's life cycle |
| Description | Identification, planning of execution, assignment and contingency planning of all security related task during the development, deployment and operation of the product/service. This includes the description of the tasks (typically referring to the process element descriptions), their scheduling, their assignment to roles and individuals. The security planning should include the execution of the validation activities and the production and distribution of the related evidence. |
| Output | Security plan for the product/service |
| Validation | Check availability of security plan. Check the security plan for requirements on content and presentation. Check that the security plan is made aware / distributed to the product/service team and affected roles in the organization. |
| Occurrences | From paper-based documentation to tool-based enforcement |
| References/dependencies | PE3: The SDLC instance mainly defines which security activities need to be planned. |
| Mandatory | No |
| Application Notes | The higher the degree of automation and enforcement of the SDLC instance is, the lower are the planning needs, since more activities will be automatically scheduled or executed |

Table 8: Security Planning

## 6.5 Software Architecture

| System Architecture | |
|---|---|
| ID | PE5 |
| Name | Software Architecture |
| Purpose | Define the software architecture of the product/service |
| Description | Define the software architecture of the product/service: specify the system components, the system environment, the basic functionality of each components, (abstract) interfaces (technical and user), data flows, technologies, etc. The architecture defines the high-level design of the product/service and is the reference for all following development activities. The architecture is typically describes using block diagrams indicating software components and their relations. |
| Output | Software Architecture specification of the product/service |
| Validation | Check availability of software architecture specification. Check the software architecture for requirements on content and presentation, e.g., consistency and completeness |
| Occurrences | From paper-based documentation using common/standardized specification techniques (e.g., TAM or UML) to formal specifications and proofs |
| References/dependencies | None |
| Mandatory | No |
| Application Notes | SAP's Technical Architecture Modelling (TAM) [35] is an example for a description technique for software architecture. |

Table 9: System Architecture

## 6.6 Threat Modelling

| Threat Modelling | |
|---|---|
| ID | PE6 |
| Name | Threat Modelling |
| Purpose | Identify technical security risks on the abstraction level of the system architecture |
| Description | The software architecture is analyzed for technical risks, i.e. potential threats imposed by an attacker. Threat modelling includes the identification of potential attack paths in the architecture, the risk level of components of the architecture based on the assets or data processed, stored or managed by the component, the identification of the product/service's attack surface and other related activities. |
| Output | Identification and analysis of threats on architecture level and proposal for mitigating the related risk, compiled in a threat modelling report or tool. |
| Validation | Availability of threat modelling report; check of the report for quality and consistency. |
| Occurrences | From paper-based to tool-based documentation, tracking of identified threats throughout the SDLC, informal, semi-formal or formal methods for threat analysis |
| References/dependencies | PE5: The software architecture is the main input for the threat modelling |
| Mandatory | No |
| Application Notes | There are a number of threat modelling methodologies and tools available that can be used in the application of this process element. [36] gives a high-level overview of some of them.<br>To reduce the entry barrier for process-based certification, we do not require threat modelling to be mandatory for an SDLC instance, but highly recommend to include it, since it is the basis for the security analysis of the software and hence, for evaluating the adequacy of the security mechanism of the product/service. |

Table 10: Threat Modeling

## 6.7 Security Functional Requirements Definition

| Security Functional Requirements Definition | |
|---|---|
| ID | PE7 |
| Name | Security Functional Requirements Definition |
| Purpose | Specify the security functionality and the security mechanisms for the product /service. |
| Description | The security functional requirements describe the technical security mechanisms that are to be implemented in the product/service or to be consumed from components or service from the environment of the product or service. They become part of the product/service's backlog and, like functional requirements, are traced and validated throughout the development of the product/service. If PE6 is part of the SDLC instance, Security Functional Requirements Definition includes a justification that the security functional requirements mitigate the treats identified throughout the threat modelling. |
| Output | Security functional requirements specification |
| Validation | Content of the product backlog |
| Occurrences | From paper-based descriptions to formal specifications |
| References/dependencies | PE6: If threat modelling is performed, then its results imply needs for security functional requirements specification, since the security functionality of the product/service is implementing (parts of) the risk mitigations identified through threat modelling |
| Mandatory | No |
| Application Notes | Examples of security mechanisms consumed from components or service from the environment of the product or service include using an external Identity provider, a secure storage service, or an open source implementation of TLS. |

Table 11: Security Functional Requirements Definition

## 6.8 Secure Programming Guidelines

| Secure Programming Guidelines | |
|---|---|
| ID | PE8 |
| Name | Secure Programming Guidelines |
| Purpose | Describe and apply state-of-the-art secure programming practices |
| Description | Over the past years, a common understanding of best practices for secure programming has emerged, intended to help developers to avoid common vulnerabilities and security pitfalls when designing and writing code. Such practices include, for instance, the use of sanitization functions, the use of standard libraries for critical security functionality like encryption or protocols, a security testing friendly code structure, and more.<br>PE8 includes the description of such practices for the organization (which can be done by reference to published guidelines) and their application to the development of the code of the product/service under evaluation. |
| Output | Documentation of the guidelines, procedures for disseminating the guidelines to developers. |
| Validation | Check availability and content of the guidelines, collect evidence for their application (e.g., by spot checks on the code or interview with developers) |
| Occurrences | Paper-based documentation of the guidelines, tool support, tool-based enforcement |
| References/dependencies | PE9: If security code analysis is performed, it provides additional evidence for the application of the secure programming guidelines. |
| Mandatory | No |
| Application Notes | Many guidelines for secure programming have already been published and can be used for defining the organization's guidelines or as a reference for them. [37-42] are examples from organizations/associations, universities and industry.<br>Training and awareness programs for developers accompany this process element on an organizational level. |

Table 12: Secure Programming Guidelines

## 6.9 Code-level Security Analysis

| Code-level Security Analysis | |
|---|---|
| ID | PE9 |
| Name | Code-level Security Analysis |
| Purpose | Find and fix security vulnerabilities by analysing the source code |
| Description | The source code produced by the organization in the development of the product/service is analyzed for security vulnerabilities. The analysis is static, i.e., the code is not executed or tested during this activity, and the foundation for the analysis is an abstraction of the program semantics, like an abstract syntax tree, a control flow graph or a data/information flow graph. Found vulnerabilities must be fixed, i.e., providing the fix becomes a new backlog item |
| Output | Analysis report including a documentation of the vulnerabilities found, product backlog items |
| Validation | Reports created, tool logs, product backlog updates |
| Occurrences | From manual code reviews over automated tools to formal analysis |
| References/dependencies | PE8: If this process element is applied, its results can be used for validating compliance with secure programming guidelines<br>PE10. Static and dynamic approaches for vulnerability analysis complement each other, hence, using both typically increases the likelihood of detecting vulnerabilities<br>PE11: Dur to their automation, static code analysis tools are also suitable for the security analysis of Open Source Software or other 3rd party components (provided their source code is available) |
| Mandatory | No |
| Application Notes | Tools for automated static code analysis are available on the market. [43] provides a list of static code analysis tools with many security focused ones among them.<br>In general, code-level security analysis focuses on known vulnerabilities or vulnerability types, but some of the techniques and tools might also be used for finding new ones.<br>Tools provided by the SPARTA CAPE program ([44] Section 3.1) are first class candidates to be used in the application of this process element. |

Table 13: Code-level Security Analysis

## 6.10 Security Testing

| Security Testing | |
|---|---|
| ID | PE10 |
| Name | Security Testing |
| Purpose | Find and fix security vulnerabilities by testing the product/service |
| Description | Perform security testing against the product/service. This includes traditional functional testing for the implementations of the security functional requirements as well as the application of dedicated dynamic security analysis and test methods and tools for the full code base in order to find known vulnerabilities. A test coverage analysis should be included in the element. |
| Output | Test plan, test cases, test protocols, test results, tool logs and reports, product backlog items |
| Validation | Sufficient coverage based on the provided coverage analysis, no failed tests |
| Occurrences | From manual testing to fully automated tools |
| References/dependencies | PE9. Static and dynamic approaches for vulnerability analysis complement each other, hence, using both typically increases the likelihood of detecting vulnerabilities<br>PE11: Due to their automation, some dynamic code analysis tools (e.g., fuzz testers) are also suitable for the security analysis of Open Source Software or other 3rd party components |
| Mandatory | No |
| Application Notes | Tools provided by the SPARTA CAPE program ([44] Section 3.1) are first class candidates to be used in the application of this process element. |

Table 14: Security Testing

## 6.11 Security Assessment of 3rd Party / Open Source Software

| Security Assessment of 3rd Party / Open Source Software | |
|---|---|
| ID | PE11 |
| Name | Security Assessment of 3rd Party / Open Source Software |
| Purpose | Assess 3rd party and OSS components with respect to their security (known vulnerabilities) and the assumptions made on them |
| Description | Analyse the security of the 3rd party components and OSS software used by the product/service. This includes the analysis of security related information available about these components (like, for instance, a security certificate, results of approval procedures, security properties being part of an SLA or any other evidence of assurance that might be accessible), including an argument that this information can be used in a compositional reasoning about the security of the product/service, as well as independent analyses by the organization developing the product/service, like black-box security testing (for components where source code is not accessible) or open source security scanners.<br>PE11 also includes the validation of assumptions made on the operational environment, e.g., resulting from PE6 |
| Output | Test and tool reports with positive verdict, approval decision |
| Validation | Availability and analysis of the reports |
| Occurrences | From code review to automated static analysis tools, from manual testing to fuzz testing, open source security scanners, any combination thereof |
| References/dependencies | PE6: Threat Modelling can lead to security assumptions on architectural components provided by 3rd parties or OSS which need to be checked in PE11<br>PE7: Security Functional Requirements can be imposed on 3rd party or OSS components that provide the required security functionality. In PE11, it needs to be checked if the requirements are met by the components chosen. |
| Mandatory | No |
| Application Notes | Tools used for code level security analysis (PE9) and security testing (PE10) might also be applicable here, depending on the depth of accessible information (black-box, specifications, source code).<br>Commercial offers for open source security scanners are available on the market, see [45] for an overview. Some of the tools provided by the SPARTA CAPE program ([44] Section 3.1) are first class candidates to be used in the application of this process element. |

Table 15: Security Assessment of 3rd Party / Open Source Software

## 6.12 Assessment of the Operational Environment

| Assessment of the Operational Environment | |
|---|---|
| ID | PE12 |
| Name | Assessment of the Operational Environment |
| Purpose | Analyse and address the security implications of the expected or actual operational environment of the product/service |
| Description | The assumed or actual operational environment of the product/service can impact the security, in particular if security functional requirements are expected to be satisfied by the environment (for example, Identity and Access Management functionality). This process element includes the analysis of the security impact of the operational environment, including the validation of assumptions made, the impact of security guidance for the environment, and more.<br>PE12 also includes the validation of assumptions made on the operational environment, e.g., resulting from PE6 |
| Output | Documentation of the analysis result and requirements matching, user guidance |
| Validation | Availability of the documentation, check for content and presentation, positive results |
| Occurrences | Depending on the specifics of the operational environment, for instance, if it is owned or controlled by the provider of the product/service, if it is actual or assumed, or if it shows other specifics (e.g., the cloud) |
| References/dependencies | PE6: Assumptions or requirements for the environment resulting from threat modelling need to be validated<br>PE20: In case of assumed operational environments and assumptions or requirements on it, the user guidance must include their descriptions and any activities required by the user to establish or maintain security. |
| Mandatory | No |
| Application Notes | In case of a cloud service deployed on a 3rd party cloud infrastructure, for instance, via a hyperscaler, this process element can translate into requirements on the security related service level agreement clauses. |

Table 16: Assessment of the Operational Environment

## 6.13 Development Environment

| Development Environment | |
|---|---|
| ID | PE13 |
| Name | Development Environment |
| Purpose | Minimizing the risk stemming from the development environment, including facilities, personnel, tools, and IT infrastructure |
| Description | Analyse the development environment (facilities, personnel, tools, IT infrastructure) for security risks and establish measures to mitigate them. Ensure the protection of corporate networks, infrastructures and resource, protect data assets, ensure the integrity of the development and deployment environment, perform security analysis of the development tools and build infrastructures and processes, ensure version control and configuration management |
| Output | Documentation of the security measures applied to facilities and personnel, security analysis of the development, deployment IT infrastructure including tools and processes |
| Validation | Availability of documentation and security analysis results |
| Occurrences | From documentation of the procedures applied to formal site inspections and certified personnel, from baseline protection of the development ICT infrastructure to high-security, isolated infrastructures |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | This is an organizational level process element and would typically only be evaluated on organizational level. Cf. Section 5.5. Exceptions might occur for high-sensitivity products/services, e.g., classified ones. |

Table 17: Development Environment

## 6.14 Vulnerability Analysis

| Vulnerability Analysis | |
|---|---|
| ID | PE14 |
| Name | Vulnerability Analysis |
| Purpose | Identify and fix security vulnerabilities in the product/service |
| Description | Perform penetration testing to identify and fix security vulnerabilities in the product/service. Penetration testing aims at gaining control over the product/service by finding and exploiting known or unknown vulnerabilities, using “hacking” tools and the specific expertise of the penetration tester. The scope of the execution of the tests is both the product/service as well as its operational environment, e.g., in order to find ways to bypass the product/service’s security functionality by exploiting vulnerabilities of the environment (cloud infrastructures, data bases, operating systems, networks, etc.) |
| Output | Test protocols, test results, product backlog items |
| Validation | Reports created, tool logs, product backlog updates |
| Occurrences | In-house penetration testing, commissioned 3rd party penetration testing with varying effort or investment, bug bounty programs, |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | -- |

Table 18: Vulnerability Analysis

## 6.15 Continuous Vulnerability Checks

| Continuous Vulnerability Checks | |
|---|---|
| ID | PE15 |
| Name | Continuous Vulnerability Checks |
| Purpose | Identify and fix security vulnerabilities during the operation of the product/service |
| Description | Vulnerability assessment is a continuous task, since new vulnerabilities might become known after deployment and during operation, the technology context might change, or updated components might introduce new vulnerabilities. This process element ensures that vulnerability analysis is executed as a continuous activity throughout the life cycle of the product/service. It includes both repeated penetration testing as well as continuous scans for known vulnerabilities and inclusion of related updates. |
| Output | Test protocols, test and scan results, product backlog items for upcoming releases (in DevOps environments) or triggering patch release processes (otherwise) |
| Validation | Reports created, tool logs, product backlog updates, availability of patches |
| Occurrences | In-house penetration testing, commissioned 3rd party penetration testing with varying effort or investment, bug bounty programs, open source security scanners |
| References/dependencies | PE14: tools and methodologies of PE14 can also be used here<br>PE16: patch releases and processes or product/service updates can be triggered by the outputs of this process element |
| Mandatory | No |
| Application Notes | This process element is basically an extension of PE11 and PE14 to the operation life cycle phase. |

Table 19: Continuous Vulnerability Checks

## 6.16 Patch / Update Processes

| Patch / Update Processes | |
|---|---|
| ID | PE16 |
| Name | Patch / Update Processes |
| Purpose | Implement a systematic approach to the provision and dissemination of security updates for the product/service |
| Description | Define a process for the provision and the management of security patches that ensures that security patches are provided and delivered in due time and customers are appropriately informed. Define KPIs on organizational level against which the process execution can be measured. |
| Output | Process description, KPIs for process execution |
| Validation | Availability and content of the documentation of the process, instantiation of procedures and tools required, KPI measurements |
| Occurrences | From documented process to tool-based enforcement |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | For cloud services or DevOps environment with release cycles shorter than the identified KPIs for patch provision, the process can be based on feeding back detected security issues in the product backlog with adequate priorities. This does not affect the requirements for customer information. |

Table 20: Patch / Update Processes

## 6.17 Secure Configuration by Default

| Secure Configuration by Default | |
|---|---|
| ID | PE17 |
| Name | Secure Configuration by Default |
| Purpose | Ensure that the product/service is delivered with the security configuration set to high security standards. |
| Description | Products or services typically allow customers to personalize security settings. In the initial state of the product/service, these configurable settings should be instantiated with default values that prioritize security over convenience, i.e., use the respective values with the highest security level. |
| Output | Guidelines for developers, configuration files |
| Validation | Approval procedure for configurations, spot checks on configuration files |
| Occurrences | From manual procedures to tool-based enforcement |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | -- |

Table 21: Secure Configuration by Default

## 6.18 Secure Deployment

| Secure Deployment | |
|---|---|
| ID | PE18 |
| Name | Secure Deployment |
| Purpose | Protecting the integrity of the code of the product/service during delivery and deployment |
| Description | Comprises activities designed to ensure that the security and the integrity of the product/service is maintained during deployment and delivery. This includes the demonstration that the product/service deployed is the same that has been produced during development and has not been tampered with (e.g., by being deployed via a secure build pipeline) |
| Output | Documentation of the process and the technical infrastructure for delivery / deployment (build pipeline, download, etc.) |
| Validation | Availability and content of the documentation, spot checks |
| Occurrences | From manual procedures to automated build pipelines and remote attestation |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | We do not consider traditional delivery via storage media (e.g., a CD or DVD), since they do not play a role in the commercial software market any more. |

Table 22: Secure Deployment

## 6.19 Formal Modelling and Analysis

| Formal Modelling and Analysis | |
|---|---|
| ID | PE19 |
| Name | Formal Modelling and Analysis |
| Purpose | Apply strong mathematically based methods for the development of the software and its security analysis |
| Description | Application of mathematically based models and methodologies in development related steps, including architecture, specification, design, implementation. Formal methods are the strongest approaches available for investigating into properties of these artefacts, like correctness, consistency, completeness, and more, since they produce mathematical proofs for the validity of the properties. Since many methods apply to abstractions of the product/service (models like security models or security policy models) the results of the application of formal methods also only apply to the models. Hence, they give the strongest possible evidence, but do not necessarily provide guarantees on the properties or the behavior of the product/service. |
| Output | Formal models, proofs |
| Validation | Availability of artefacts and proofs, proof checks |
| Occurrences | Variations on depth (from abstract security model to code verification), strength (expressiveness of the formal language), degree of automation, tool support. Many approaches and tools exist. Chapter 2 of [46] can be used for a first overview. |
| References/dependencies | Formal methods will most likely be applied in the context of PE5, PE6, PE7, PE9 |
| Mandatory | No |
| Application Notes | Recommended only for high-risk environments and basically introduced to support extension to product-centric certification, e.g. following the CC. Cf. Chapter 8 |

Table 23: Formal Modelling and Analysis

## 6.20 Tools and Automation

| Tools and Automation | |
|---|---|
| ID | PE20 |
| Name | Tools and Automation |
| Purpose | Define and identify the tools and automated process applied in the SDLC |
| Description | Application of process elements can be improved in quality and accelerated by using tools for the support or automation of their execution. We list tools and automation as a separate process element, because methodologies for automation typically can span several process elements (e.g., for tracking requirements and tasks) and may be selected and maintained centrally by the organization rather than in the scope of an individual process element. |
| Output | Description of tool landscape, application of tools in process elements of the SDLC instance |
| Validation | Tool logs and outputs |
| Occurrences | Tools for process management, software development, software build and deployment, secure operations, etc. |
| References/dependencies | Applicable to all process elements |
| Mandatory | No |
| Application Notes | -- |

Table 24: Tools and Automation

## 6.21 User Guidance

| 1.1 User Guidance | |
|---|---|
| ID | PE21 |
| Name | User Guidance |
| Purpose | Describe obligations for the user for securely operating the product/service |
| Description | The product/service may leave deployment and operation options to the customer or its users (e.g., definition and assignment of roles, authentication methods, configuration of protocols and network parameters, use of own encryption libraries, data sharing preferences, etc.). Since the security of the product/service may depend on the choices the customer make, they need to be made aware of the assumptions that have been made on their actions for the evaluations, and the obligation they have to meet to maintain the claims of the certificate. These assumptions and obligations are described in a guidance document and handed over to the customer. |
| Output | User guidance document, hand-over/dissemination procedure |
| Validation | Availability and content of the document, hand-over protocol |
| Occurrences | Documented guidelines, can be supported by a tool for the customer's use |
| References/dependencies | -- |
| Mandatory | No |
| Application Notes | -- |

Table 25: User Guidance

# Chapter 7 Assurance Levels

*Process elements are grouped into meaningful sets that might form the basis of an assurance level definition, based on the characteristics and the hierarchy defined in the previous chapter.*

Assurance levels are introduced to certification to allow to adjust the evaluation effort and strength for an individual certification project, so that they can respond to different results of risk assessment for the product or service at hand. The higher the risk, the more and stricter the requirements on evaluation (including both vendor activities to provide evidence and evaluator activities to analyse the evidence) are. Assurance levels built into a certification scheme can also serve a strategic purpose, by low assurance levels providing a low entry barrier for vendors into certification, from which higher assurance levels can be incrementally achieved.

The introduction of assurance levels is not mandatory for certification schemes. If a scheme does not define them, it basically responds to one single assurance level which is determined by the evaluation criteria of the scheme. However, introducing assurance levels allows certification schemes to scale to different levels of risk exposure.

In its Article 52, the EU-CSA introduces three assurance levels “basic”, “substantial” and “high” which are defined as follows:

- “A European cybersecurity certificate or EU statement of conformity that refers to assurance level ‘basic’ shall provide assurance that the ICT products, ICT services and ICT processes for which that certificate or that EU statement of conformity is issued meet the corresponding security requirements, including security functionalities, and that they have been evaluated at a level intended to minimise the known basic risks of incidents and cyberattacks. The evaluation activities to be undertaken shall include at least a review of technical documentation. Where such a review is not appropriate, substitute evaluation activities with equivalent effect shall be undertaken.
- A European cybersecurity certificate that refers to assurance level ‘substantial’ shall provide assurance that the ICT products, ICT services and ICT processes for which that certificate is issued meet the corresponding security requirements, including security functionalities, and that they have been evaluated at a level intended to minimise the known cybersecurity risks, and the risk of incidents and cyberattacks carried out by actors with limited skills and resources. The evaluation activities to be undertaken shall include at least the following: a review to demonstrate the absence of publicly known vulnerabilities and testing to demonstrate that the ICT products, ICT services or ICT processes correctly implement the necessary security functionalities. Where any such evaluation activities are not appropriate, substitute evaluation activities with equivalent effect shall be undertaken.
- A European cybersecurity certificate that refers to assurance level ‘high’ shall provide assurance that the ICT products, ICT services and ICT processes for which that certificate is issued meet the corresponding security requirements, including security functionalities, and that they have been evaluated at a level intended to minimise the risk of state-of-the-art cyberattacks carried out by actors with significant skills and resources. The evaluation activities to be undertaken shall include at least the following: a review to demonstrate the absence of publicly known vulnerabilities; testing to demonstrate that the ICT products, ICT services or ICT processes correctly implement the necessary security functionalities at the state of the art; and an assessment of their resistance to skilled attackers, using penetration testing. Where any such evaluation activities are not appropriate, substitute activities with equivalent effect shall be undertaken.”

While the EC-CSA assurance levels directly refer to the risk (“known basic risks”, “actors with limited skills and resources”, “actors with significant skills and resources”, the Common Criteria define assurance levels based on the scope, depth and rigor of the assurance requirements. Table 1 in

Section 3.1 introduces the CC assurance levels and their characterization based an analysis strength.

We aim at defining assurance levels for the process-based approach. However, the full specification of assurance levels is beyond the scope of this document and left to the further work. In the remainder of this chapter, we discuss the dimensions along which assurance levels for the process-based approach can be defined. These dimensions indicate which options a vendor has to adapt their SDLC instance in terms of the process steps which application is to be enforced and the strictness of their occurrences and validation. In case of the latter, validation can be performed by the vendor or the evaluator, in analogy to stronger CC assurance components requiring independent evaluator actions.

Following this approach, we end up with three relevant dimensions: the process element selection, the occurrences of the process elements applied, and the validation actions performed

**Dimension 1: process element selection**

The process-based approach is designed in a way that the actual instance of the SDLC applied to a product/service can vary according to the risk assessment. Mandatory process elements PE2 and PE3 are implementing the risk assessment and the selection of process elements for the given project. Note that all other process elements of Chapter 6 are not mandatory, so that any combination of them respecting the indicated dependencies would be allowed in principle. However, not all combinations are meaningful and the validation activities of PE2 and PE3 take care of selecting only meaningful ones. Assurance levels can prescribe particular SDLC instances based on standardized risk situations.

If this dimension is chosen for the definition of assurance levels, PE2 would be reduced to a justification of the applicability of the standardized risk situation, and PE3 would be reduced to the determination of an assurance level. In addition, the organization might not have the full freedom anymore with respect to their choices of implementing the process elements in their library.

**Dimension 2: process element occurrences**

The process element catalogue in Chapter 6 describes process elements on a general level only, in order to allow an organization to be flexible with respect to their implementation. The possible occurrences of the process elements differ in their scope, depth and rigor. For instance, security testing (PE10) can be done manually or automated, with proof of coverage or without, on interface or code level, and more. Assurance levels can refer to defined (sets of) process element occurrences.

If this dimension is chosen for the definition of assurance levels, the organization might not have the full freedom anymore with respect to their choices of implementing the process elements in their library.

**Dimension 3: process element validation**

Validation activities defined for process elements can range from simple checks for process application (for instance, the existence of a document or a log file) up to a thorough analysis of the process element output's content. Clearly, these different activities lead to different strength of assurance and can be used for assurance level definitions.

The indication of the validation activities in the process element description does not include a determination of who is executing the validation activity. Hence, it is open to the inclusion of independent evaluator actions for higher assurance levels.

If this dimension is chosen for the definition of assurance levels, it can have an impact on the fulfilment of the requirements stated in Chapter 4. Scalability, agility and economic viability depend on the majority of the evaluation burden put into the definition and selection of the SDLC instance

rather than evaluating the details of the process element application and their outcomes. Assurance level definitions based on this dimensions should therefore be carefully done with the requirements taken into account.

On the other hand, assurance level definitions based on the validation activities open extension paths to product-centric certification and described in Chapter 8.

In summary, assurance levels for a process-based certification scheme can be referring to any subset of the above dimensions, and any combination of their values. An assurance level definition would need to identify meaningful combinations, taking the requirements of Chapter 4 into account.

# Chapter 8 Compatibility with Common Criteria

*This chapter defines a migration path from process-centric to product-centric evaluation of a product or a service. In particular, it explains how process elements evaluated in a process-centric scheme can be used to support a Common-Criteria-style product certification.*

Process-based certification complements CC-like product certification in the sense of providing an effective certification scheme for highly dynamic environments like the cloud, and by paving the way for high assurance level CC certification in cases of particular security sensitivity. In this chapter, we want to demonstrate by means of example how process-based certification can be transformed into product-based certification by strengthening occurrences of process elements and increasing the requirements on rigor for the validation of process element application.

The CC assume that the development of the ToE is following a process that leads to the production of documents or other artefacts that are examined by the evaluator for presentation and content. In addition, independent activities by the evaluator are required that include activities that are not part of the vendor's processes or that are replacing activities that otherwise would be (or already have been) executed by the vendor.

In fact, all assurance classes of the CC, with the exception of APE (Protection profile evaluation), ACE (Protection profile configuration evaluation), ASE (Security target evaluation), the family ATE_IND (Independent testing) and AVA (Vulnerability assessment) are referring to outputs of certain process elements of the development process. That includes the compositional assurance classes, since they refer to the specification, analysis and testing of the composed ToE, i.e., tasks that are part of an SDLC.

An organization's process element library therefore can contain process elements whose outputs meet CC requirements or introduce additional process elements or occurrences of existing process elements for those outputs that cannot yet be produced (for instance, a security model, an effectiveness analysis of the security functionality or a formal verification). Following an application risk analysis (which maps to a Security Target or a Protection Profile), these process elements can be selected for the given application task and used in a later formal CC evaluation and certification effort, where the outputs and validation results of the process elements are evaluated.

The general approach would then be as follows:

a) Map the required documents and artifacts to process elements in the organization's process element library.
b) If the CC requires documents or artefacts that are not produced by existing process elements in the library or by the existing occurrences of the process elements of the library, introduce additional process elements from the catalogue in the organization's process element library[5], or additional occurrences to the process elements of the library, or additional or strengthened validation activities to the process elements of the library.
c) Introduce additional validation elements that require the analysis of the process outputs according to the criteria of the CC (in addition to the validation activities of the process-based approach which focus on the evidence of the application of the process elements like the existence of documents or log entries, but not on the content of the outputs themselves).
d) Add an occurrence to PE2 "Product/Service Risk Assessment" that refers to the provision of a CC security target or a protection profile conformance claim. This occurrence includes the requirements from the security target related assurance families, with the exception of those

[5] If CC would require outputs that are not produced by any process element that is in the catalogue (cf. Chapter 6), these process elements would need to be added to the catalogue and require an update on scheme level. However, we do not see this need for the assurance components of CC v3.1R5, as those can be introduced as process occurrences of existing elements of the catalogue, as extensions of the required outputs of the application of a process element, or as strengthened validation activities for a process element.

of ASE_ECD ”Extended components definition”, which go into PE5 “Software Architecture”, and ASE_REQ “Security requirements”, which (partially) go into PE6 “Threat Modeling” and PE7 “Security Functional Requirements Definition”, respectively.

e) Set up the SDLC instance as output of PE3 “SDLC Instance Definition” to include all CC relevant process elements, with the CC-driven occurrences and the additional validation requirements referring to the CC requirements.
f) Execute the SDLC instance for the development of the product or service.
g) Independent evaluator activities can be executed in parallel with the development (with the advantage of having certification strongly integrated and minimizing the delay between development completion and certificate issue) or at any time after development.

Of course, such a “CC instance” of the SDLC would not share the benefits of agility and scale the process-based approach would provide. But it can be seen as a straightforward transformation path from process-based to product-based evaluation and certification. An organization that has set up a security framework for process-based certification as described in this document, can maintain their investments and security practices, and extend them rather than replace them if they would like to go for full CC certification.

To illustrate the feasibility of extension of process elements, we look into an example. The assurance class ATE “Tests” includes families and components aiming at the confirmation that security functions behave according to their specification. Figure 6 shows its decomposition into families related to coverage, depth, functional test and independent testing.

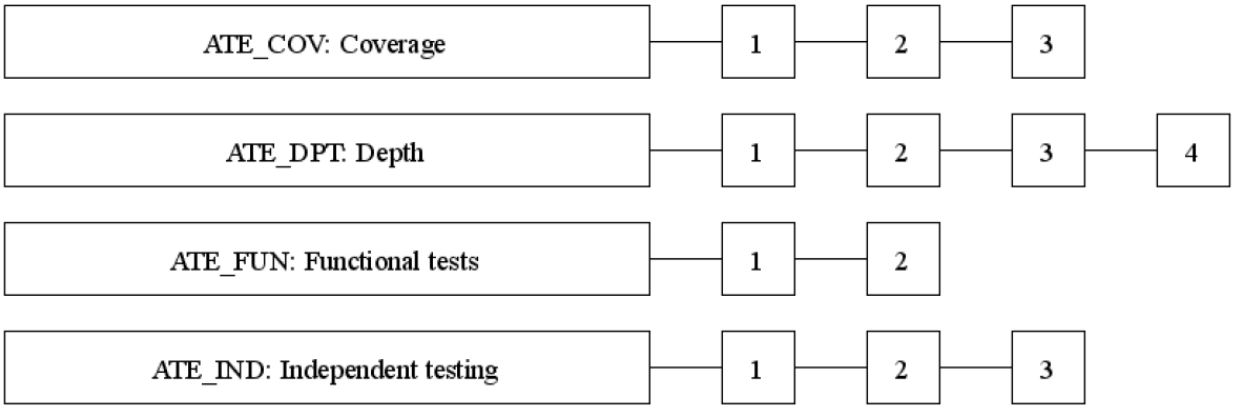


Figure 6: CC Test class decomposition (from [8])

The assurance components related to tests are indicated in Figure 6 and comprise of the components listed in Table 26.

| Assurance family | Assurance components |
|---|---|
| ATE_COV Coverage | ATE_COV.1 Evidence of coverage |
| | ATE_COV.2 Analysis of coverage |
| | ATE_COV.3 Rigorous analysis of coverage |
| ATE_DPT Depth | ATE_DPT.1 Testing: basic design |

| | |
|---|---|
| | ATE_DPT.2 Testing: security enforcing modules |
| | ATE_DPT.3 Testing: modular design |
| | ATE_DPT.4 Testing: implementation representation |
| ATE_FUN Functional tests | ATE_FUN.1 Functional testing |
| | ATE_FUN.2 Ordered functional testing |
| ATE_IND Independent testing | ATE_IND.1 Independent testing - conformance |
| | ATE_IND.2 Independent testing - sample |
| | ATE_IND.3 Independent testing - complete |

Table 26: ATE assurance components

Process elements related to testing (PE10[6]) can be extended to cover the assurance components of the ATE class. Table 27 shows what would be needed for this extension

| **Assurance components** | |
|---|---|
| ATE_COV.1 | Analysis of test coverage is included in PE10 validation activities, refine these activities with respect to evidence |
| ATE_COV.2 | Analysis of test coverage is included in PE10 validation activities, refine these activities with respect to analysis |
| ATE_COV.3 | Analysis of test coverage is included in PE10 validation activities, refine these activities with respect to rigorous analysis |
| ATE_DPT.1 | Refine the PE10 process element occurrences and validation activities with respect to basic design |
| ATE_DPT.2 | Refine the PE10 process element occurrences and validation activities with respect to security enforcing modules |
| ATE_DPT.3 | Refine the PE10 process element occurrences and validation activities with respect to modular design |
| ATE_DPT.4 | Refine the PE10 process element occurrences and validation activities with respect to implementation representation |
| ATE_FUN.1 | Refine the PE10 process element occurrences and validation activities with respect to functional testing |
| ATE_FUN.2 | Refine the PE10 process element occurrences and validation activities with respect to ordered functional testing |

[6] Note that ATE does not include penetration testing, hence, PE14 and PE15 are not included

| Assurance components | |
|---|---|
| ATE_IND.1 | Introduce a new validation activitiy to PE10 referring to independent conformance testing |
| ATE_IND.2 | Introduce a new validation activitiy to PE10 referring to independent sample testing. This requires a process element occurrence that includes test case specification. |
| ATE_IND.3 | Introduce a new validation activitiy to PE10 referring to independent complete testing. This requires a process element occurrence that includes test case specification. |

Table 27: Process element extensions for ATE assurance components

The specification of assurance components in the CC includes the identification of dependencies to other assurance components, Since the process element specification supports dependency definitions, the CC dependency specifications can be resolved by adding matching dependencies to the process element specification.

# Chapter 9 Analysis

*This chapter includes an indicative analysis of the strength of the security claims based on a process-centric evaluation compared to a product-centric evaluation. This is not a formal or strong analysis, but rather a reference to common experiences and guidelines.*

In this Chapter, we analyse if the approach to cybersecurity certification for modern commercial software products or systems described in this deliverable is reasonable and forms a good basis for such a certification scheme. The analysis consists of the following parts:

- An analysis of the proposed approach with respect to the requirements stated in Chapter 4;
- A discussion of how a certification scheme based on the proposed approach would meet the security objectives stated in Article 51 of the EU-CSA;
- A discussion of the adequacy and completeness of the process elements proposed in Chapter 6;
- An informal reasoning about the strength of claims that can be achieved with a process-based scheme.

## 9.1 Requirements

The intent of the principles of a process-based security certification scheme for software products and services as it is outlined in this document is to create the foundation for an economically viable certification scheme. In Chapter 4, we have outlined what “economically viable” means, and stated more detailed requirements on a certification approach and scheme. In the following, we revisit these requirements and argue how they are met by the proposed approach. For the reader’s convenience, the requirements definitions are repeated here.

**R0: A certification scheme for modern commercial software products and services should aim at increasing the confidence in the validity of the desired security characteristics.**

The major aspect of this requirement is that does not aim at and cannot provide security guarantees or proofs. The actual claim of a certificate is that the product or service has withstood thorough examination according to a defined assurance level based on agreed-upon (published or standardized) criteria. In case of security certification, the examination produces evidence that the security requirements are adequate and met by the product/service, as far as can be told from the collected evidence. This provides some assurance for the security of the product/service and allows to put confidence in the validity of the desired security characteristics of the product/service

In our case, this claim is based on assuring the application of today’s best practices for the secure development, deployment and operation of a product or service. The mandatory process elements PE2 and PE3 lead to a thorough risk analysis being performed and to the application of an SDLC instance that is adequate to the identified risk. Intentionally, we refer to the application of the SDLC and not just its definition, since the validation activities defined for each process element provide evidence for their application and execution.

The criteria in our approach are based on process elements and do not include security functional requirements. However, the inclusion of the process elements PE6 “Threat Modeling” and PE7 “Security Functional Requirements Definition” together with the risk analysis in PE2 leads to an adequate set of security functions being specified, with the other development-related process elements PE7 to PE14 ensuring that they are implemented, tested, and no vulnerabilities have been detected.

Section 9.4 investigates closer into the differences between claims resulting from product-based and process-based certification approaches.

**R1: A certification scheme for modern commercial software products and services must allow to certify complex systems with rich dependencies, large size and limited control of the ToE owner/vendor. It has to take into account that essential security functionality (for instance, Identity and access management) is outsourced, i.e., not part of the ToE, and provided by a 3rd party (for instance, the cloud infrastructure provider).**

In general, a process-based approach meets this requirement by focusing on the procedures and activities performed when using 3rd party contributions. Therefore, and by default, it does not require full control over those contributions, as would be needed for, e.g., accessing development documentation or 3rd party source code. The processes elements can be selected and instantiated in a flexible way, reflecting the situation at hand.

Our approach addresses this requirement by including both process elements related to software developments of the provider itself as well as to software components provided by 3rd parties or open source communities used in the product/service's software. Threat modeling involves the full scope of the product/service (including foreign components) and may lead to assumptions of the security properties of or the security functionality provided by the foreign components, which are validated in PE11 and / or PE12.

Process elements PE11, PE12, PE14, PE16 include dedicated activities related to 3rd party or OSS components and well as outsourced security functionality.

**R2: A certification scheme for modern commercial software products and services should support a risk-based approach to certification, with the elements for protection and assurance (process elements, evidence required and vendor and evaluator actions defined) being selected based on a risk assessment for the individual ToE. To allow comparisons, sets of elements considered to be adequate for typical risk postures can be combined into assurance levels.**

The approach presented in this document follows this requirement directly by mandating a risk analysis in PE2 and a product/service specific instance of the organization's SDLC following the results of the risk analysis in PE3. It supports the definition of assurance levels, cf. Chapter 7.

**R3: A certification scheme for modern commercial software products and services should support the full life cycle of a software product or service, including its operation. Assurance elements relating to the maintenance of the security of the product or service, including updates and security patches, and its continuous assessment, should be included in the criteria.**

PE12, PE15-18, PE21 are those process elements that explicitly address life cycle phases beyond the development of the product/service, They refer to the operational environment (PE12), the deployment of the product/service (PE17 and PE18), the maintenance of the security of the software throughout its life cycle (PE15 and PE16) and everything that is required from the user to operate and use the product/service securely.

Updates and security patches are covered by PE16, continuous assessment of the security is included in PE15.

**R4: A certification scheme for modern commercial software products and services should support the certification of composed systems using components, services and infrastructures from foreign entities or OSS, including those based on components that are not certified themselves. In case of using non-certified components, the scheme must**

**provide assurance elements justifying that those components do not impact the security of the ToE, or that, in case of components providing security functionality, there is evidence that the provided functions are functionally effective and correctly implemented.**

The assumption that products or services are built in a compositional way, using components developed by the product/service vendor as well as those provided by 3rd parties including OSS is designed into the approach, and process elements refer to both in-house development (PE8 – PE10) as well as external component usage (PE11). PE11 can be instantiated depending on the security information and evidence available for the component, ranging from re-using a certificate and its related documentation for the component to the conduction of security tests (black-box, fuzzing), the application of security scanning tools (if the source code is available, like it is the case for OSS) to vulnerability analysis of the components (e.g., penetration testing). PE11 includes the analysis of potential restrictions applying to compositionality of security.

**R5: A certification scheme for modern commercial software products and services should provide the necessary agility to not impact the release frequency and speed of modern commercial software systems including releases stemming from fully automated build processes. The criteria should allow for the certification of a continuous series of releases that are provided over time with a minimum of additional effort per release (ideally, in an automated fashion).**

This requirement is met by focusing the security evaluation on the processes, which are expected to change only occasionally, by the separation of evaluation activities on organizational and product/service level, with the activities requiring more significant effort being performed on organizational level, and by making the reasonable assumption that the risk assessment and, hence the appropriate SDLC instance for a series of releases of the same product/service do not significantly change, in general. The latter assumption is justified by viewing an agile development project as resulting in a series of releases that differ in terms of the features enabled. The set of those features is largely known at the beginning of the project (though not yet developed) and can be included in the risk assessment (PE2), the system architecture (PE5) and the threat model (PE6). Hence, with a high probability, the results of these process elements are the same for all or most releases. The same applies for the SDLC instance (PE3), which would not need to be changed if the risk assessment doesn't.

The effort related to the set-up and the evaluation of the organizational security framework (PE1) is on organizational level and it can be safely assumed that the organizational security framework is stable for a longer period. Within this framework, the risk assessment (PE2), the instantiation of the SDLC (PE3) and the security planning (PE4) are part of each development project for a product and service. For each individual release, the previous results of PE2 – PE4 have only to be analyzed if they still apply, in most cases the expectation is that they do. Then, the activities of PE2 – PE4 do not have to be repeated, and the process elements of the SDLC instance are applied in the further development, deployment and operation of the product/service. For the evaluation and certification, it only remains to check if they have been applied by executing the respective validation steps. Altogether, the effort per release is minimized.

The higher the usage of tools and the degree of automation (PE20), the lower the overall effort for evaluation will be. In case of lower assurance levels, where the evaluation focuses on validating the existence of the required process element output, we expect that the major port of the evaluation can be executed automatically.

**R6: The costs and efforts required for the certification of modern commercial software products and services should be limited to an amount that can be rewarded by the market.**

Only the market figures themselves, once a certification scheme following the approach described in this document is established, will tell if this requirement is met, but we consider the chance being high that this will be the case. The process elements described are part of an SDLC that every

organization that is aiming at delivering secure products or service will have to put in place. With its approach of shifting major evaluation efforts to the organizational level and, hence, keeping the additional evaluation costs per release comparatively low, the approach scales with the number of products/services and their respective releases being evaluated and certified.

## 9.2 Security Objectives of the EU-CSA

Article 51 of the EU-CSA defines “Security objectives of European cybersecurity certification schemes” and states that “A European cybersecurity certification scheme shall be designed to achieve, as applicable, at least the following security objectives (repeated from Section 2.1 for the reader’s convenience):

- (a) to protect stored, transmitted or otherwise processed data against accidental or unauthorised storage, processing, access or disclosure during the entire life cycle of the ICT product, ICT service or ICT process;
- (b) to protect stored, transmitted or otherwise processed data against accidental or unauthorised destruction, loss or alteration or lack of availability during the entire life cycle of the ICT product, ICT service or ICT process;
- (c) that authorised persons, programs or machines are able only to access the data, services or functions to which their access rights refer;
- (d) to identify and document known dependencies and vulnerabilities;
- (e) to record which data, services or functions have been accessed, used or otherwise processed, at what times and by whom;
- (f) to make it possible to check which data, services or functions have been accessed, used or otherwise processed, at what times and by whom;
- (g) to verify that ICT products, ICT services and ICT processes do not contain known vulnerabilities;
- (h) to restore the availability and access to data, services and functions in a timely manner in the event of a physical or technical incident;
- (i) that ICT products, ICT services and ICT processes are secure by default and by design;
- (j) that ICT products, ICT services and ICT processes are provided with up-to-date software and hardware that do not contain publicly known vulnerabilities, and are provided with mechanisms for secure updates.”

Out of these, a), b) c), e), f) and h) refer to security functional requirements and, following the process-based approach, will not be explicitly required by a certification scheme based on our ideas. However, if they are applicable to the product/service, the process elements for security functional requirements (PE7) together with the security analysis and testing process elements (PE9, PE10, PE11) will ensure that they are included and implemented.

For the remaining objectives, Table 28 shows which process elements address them.

| EU-CSA objective | Process Element |
|---|---|
| (d) to identify and document known dependencies and vulnerabilities; | PE9, PE10, PE11, PE14<br>Referring to both in-house developments and foreign components consumed, comprising code analysis, testing, and vulnerability analysis |

| EU-CSA objective | Process Element |
|---|---|
| (g) to verify that ICT products, ICT services and ICT processes do not contain known vulnerabilities; | PE9, PE10, PE11, PE14<br>Referring to both in-house developments and foreign components consumed, comprising code analysis, testing, and vulnerability analysis |
| (i) that ICT products, ICT services and ICT processes are secure by default and by design; | PE6, PE8, PE11, PE17, PE18<br>Include the essentials of a secure-by-design and secure-by-default approach to software security |
| (j) that ICT products, ICT services and ICT processes are provided with up-to-date software and hardware that do not contain publicly known vulnerabilities, and are provided with mechanisms for secure updates. | PE15, PE16<br>Ensure that the product/service is continuously monitored for security vulnerabilities and that fixes to identified vulnerabilities are provided |

Table 28: Mapping of EU-CSA objectives and process elements

## 9.3 Adequacy and Completeness

The set of process elements for a process-based security certification scheme for products and services should be adequate and complete with respect to the state-of-the-art in secure software development. We investigate into this by comparing the process elements introduced in Chapter 6 with the tasks defined in the NIST white paper for a Secure Software Development Framework (SSDF) [12]. We consider [12] as a good reference, since it is based on the analysis of 18 references introducing secure development best practices or secure development life cycles (cf. Section 3.3).

In Table 29, we look into each task defined in [12] and map them to the process element, where those tasks are included in the process activities as given by the process element description. If there is an entry, the task is covered by at least one of the process elements of Chapter 6. If there is no matching process element, there would be a gap in our set, which is not the case.

Tasks in [12] and process elements in Chapter 6 differ in their abstraction level, and future work could be devoted to bring them closer together. Sometimes, process elements are more abstract – the process element is occurring in more than one row, sometimes SSDF tasks are more abstract – in the respective row there are more than one process element identified. However, we found the task level of [12] being the most appropriate to analyse our process elements in terms of adequacy and completeness.

Tasks in [12] are labelled according to the group in which they fall:

- Prepare the Organization (PO)
- Protect the Software (PS)
- Produce Well-Secured Software (PW):
- Respond to Vulnerabilities (RV)

The fist number in the label refers to the so-called practice, an intermediate level of abstraction between the groups and the tasks.

| NIST SSF Task | Process Element |
|---|---|
| PO.1.1: Identify all applicable security requirements for the organization's general software development, and maintain the requirements over time. | PE1 |
| PO.2.1: Create new roles and alter responsibilities for existing roles to encompass all parts of the SSDF. Periodically review the defined roles and responsibilities, and update them as needed. | PE1 |
| PO.2.2: Provide role-specific training for all personnel with responsibilities that contribute to secure development. Periodically review role-specific training and update it as needed. | PE1 |
| PO.2.3: Obtain upper management commitment to secure development, and convey that commitment to all with SSDF-related roles and responsibilities. | PE1 |
| PO.3.1: Specify which tools or tool types are to be included in each toolchain and which are mandatory, as well as how the toolchain components are to be integrated with each other. | PE20 |
| PO.3.2: Following sound security practices, deploy and configure tools, integrate them within the toolchain, and maintain the individual tools and the toolchain as a whole. | PE20 |
| PO.3.3: Configure tools to collect evidence and artifacts of their support of the secure software development practices. | PE20 |
| PO.4.1: Define criteria for software security checks throughout the SDLC. | Refers to the validation activities for each PE |
| PO.4.2: Implement processes, mechanisms, etc. to gather the necessary information in support of the criteria. | Process element outputs |
| PS.1.1: Store all forms of code, including source code and executable code, based on the principle of least privilege so that only authorized personnel have the necessary forms of access. | PE 13 |
| PS.2.1: Make verification information available to software consumers. | PE21 |
| PS.3.1: Securely archive a copy of each release and all of its components (e.g., code, package files, third-party libraries, documentation), and release integrity verification information. | PE18 |
| PW.1.1: Use forms of risk modeling, such as threat modeling, attack modeling, or attack surface mapping, to help assess the security risk for the software. | PE2, PE6, PE19 |
| PW.2.1: Have a qualified person who was not involved with the software design review it to confirm that it meets all of the security requirements and satisfactorily addresses the identified risk information. | Occurrence of PE9 |

| NIST SSF Task | Process Element |
|---|---|
| PW.3.1: Communicate requirements to third parties who may provide software modules and services to the organization for reuse by the organization's own software. | PE11, PE12<br>(however, the basic assumption in our approach is that in commercial software development foreign software components are taken as they are) |
| PW.3.2: Use appropriate means to verify that commercial, open source, and all other third-party software modules and services comply with the requirements. | PE11 |
| PW.4.1: Acquire well-secured components (e.g., software libraries, modules, middleware, frameworks) from third parties for use by the organization's software. | PE11 |
| PW.4.2: Create well-secured software components in-house following SDLC processes to meet common internal software development needs that cannot be better met by third-party software. | PE1, PE2, PE3, PE8<br>(essentially, the whole approach) |
| PW.4.3: Where appropriate, build in support for using standardized security features and services (e.g., integrating with log management, identity management, access control, and vulnerability management systems) instead of creating proprietary implementations of security features and services. | PE8, PE18 |
| PW.5.1: Follow all secure coding practices that are appropriate to the development languages and environment. | PE8 |
| PW.5.2: Have the developer review their own human-readable code, analyze their own human-readable code, and/or test their own executable code to complement (not replace) code review, analysis, and/or testing performed by others. | PE9, PE10 |
| PW.6.1: Use compiler and build tools that offer features to improve executable security. | PE13, PE20 |
| PW.6.2: Determine which compiler and build tool features should be used and how each should be configured, then implement the approved configuration for compilation and build tools, processes, etc. | PE13, PE17, PE20 |
| PW.7.1: Determine whether code review (i.e., a person directly looks at the code to find issues) and/or code analysis (i.e., tools are used to find issues in code, either in a fully automated way or in conjunction with a person) should be used. | PE3, PE9 |
| PW.7.2: Perform the code review and/or code analysis based on the organization's secure coding standards, and document and triage all discovered issues and recommended remediations in the development team's workflow or issue tracking system. | PE8, PE9 |

| NIST SSF Task | Process Element |
|---|---|
| PW.8.1: Determine if executable code testing should be performed and, if so, which types should be used. | PE3, PE10 |
| PW.8.2: Design the tests, perform the testing, and document the results. | PE10 |
| PW.9.1: Determine how to configure each setting that has an effect on security so that the default settings are secure and do not weaken the security functions provided by the platform, network infrastructure, or services. | PE17 |
| PW.9.2: Implement the default settings (or groups of default settings, if applicable), and document each setting for software administrators. | PE17 |
| RV.1.1: Gather information from consumers and public sources on potential vulnerabilities in the software and any third-party components that the software uses, and investigate all credible reports. | PE14 |
| RV.1.2: Review, analyze, and/or test the software's code to identify or confirm the presence of previously undetected vulnerabilities. | PE9, PE10, PE14 |
| RV.1.3: Have a team and process in place to handle the responses to vulnerability reports and incidents. | PE16 |
| RV.2.1: Analyze each vulnerability to gather sufficient information to plan its remediation. | PE14, PE15 |
| RV.2.2: Develop and implement a remediation plan for each vulnerability. | PE14, PE15, PE16 |
| RV.3.1: Analyze all identified vulnerabilities to determine the root cause of each vulnerability. | PE14, PE15 |
| RV.3.2: Analyze the root causes over time to identify patterns, such as when a particular secure coding practice is not being followed consistently | PE15 |
| RV.3.3: Review the software for other instances of the reported problem and proactively fix them rather than waiting for external reports. | PE14, PE15 |
| RV.3.4: Review the SDLC process, and update it as appropriate to prevent (or reduce the likelihood of) the root cause recurring in updates to this software or in new software that is created. | PE1, PE2, PE3 |

Table 29: Mapping of NIST SSF Tasks and process elements

## 9.4 Strength of Claims

There is no formal way to analyse the strength of claims resulting from a security certification. This is due to the fact that they are based on expert evaluation of evidence (according to defined criteria), where, even if the definition of criteria aim at levelling that, the individual assessment and interpretation of the expert evaluator influences the resulting verdict. Evaluators can also make mistakes. Even though all provisions of the scheme and of the criteria aim at minimizing this risk, it

cannot be fully eliminated. Hence, the value of a certificate is that it allows its consumer to put confidence (more or less, depending on the assurance level) in the security of the ToE, with all the provisos applying, but expressing that a thorough examination has been passed.

To approximate the assessment of the security claims resulting from a process-based scheme, we compare them with respect to those resulting from product-based schemes, like the CC.

In a product-based scheme, the product itself is analysed based on the analysis of documentation and other artefacts (code, proofs, test results) produced for that very product. Even though the CC, for instance, assume that the evidence is resulting from the application of certain processes, it is only the product specific evidence that is considered. This gives rise to claims about properties or functionalities of the specific product under evaluation. Such claims have a structure like

**“The evidence shows that the product/service has these properties, provides this functionality, etc”**

A process-based scheme introduces an indirection to the reasoning. Its claims follow the view that products can show properties or functionalities because they were produced by applying methods, tools and processes that are targeted to produce the desired properties or functionalities. The “what” is replaced by the “how”, assuming that targeted “how” leads to the desired “what”.

Claims resulting from a process-based approach to evaluation and certification have a structure like

**“The evidence shows that the product/service has been developed (, deployed and is operated) in a way that leads to having these properties, providing this functionality, etc.”**

The more specific and the stricter the processes are, the smaller is the difference between these two types of statements.

# Chapter 10 Summary and Conclusion

This deliverable describes an alternative approach to product and service certification that is suitable for agile and dynamic development environments. Instead of analysing the product itself, it is based on assessing the processes, tools and methodologies that form the secure development life cycle at the vendor's organisation, as well as on evidence that these processes are applied properly for the development of the product or service at hand. Compared to product-centric security certification approaches, the process-centric methodology has the potential to scale and cope with agility,

The document does not describe a full certification scheme, but focuses on the basic principles and components of a process-based approach to software product and service certification. First, it analyses the requirements for software product and service certification that need to be met if security certification should be attractive for a commercial market: coping with complexity, scalability and agility of today's software, supporting a risk based approach and keeping the costs of certification manageable. Second, it introduces the basic constituents of a certification approach that meets the requirements: a secure development life cycle (SDLC) and its process elements that can be tailored to individual products or services following a risk assessment, ensuring a high security quality by aligning with best security practices and requiring validation of the application of the process, a catalogue of process elements that can be instantiated and combined to a product-specific SDLC and meaningful combinations of process elements, their instantiations and their validation activities into assurance levels.

The proposal for a process-based approach for security certification is analysed in detail against the requirements stated, the state-of-the-art in secure development, deployment and operations practices, and for their support of the security objectives stated in the European Cybersecurity Act. It turns out that it is designed in a way that allows to meet the requirements, and, hence, is a good basis for a security certification scheme that is successful in a commercial environment, in terms of ease of adoption and economic viability. The approach itself is scalable and can be tuned from very basic inexpensive checks of process definition and application to rigorous evaluation. This is demonstrated by showing how the approach can be extended to support full common-Criteria-style product-centric evaluations and certifications.

Further research is needed on the way from the principal approach described in this document to an actual certification scheme based on its ideas. This includes, among others:

- Refinement of the assurance levels
- More detailed identification of process element occurrences and analysis of their respective strength
- Definitions of reference examples of SDLC instances, in particular related to the assurance levels of the EU-CSA

# Chapter 11 List of Abbreviations

| Abbreviation | Translation |
|---|---|
| ASC | Application Security Control |
| CAP | Composed Assurance Package |
| CC | Common Criteria [8] |
| CI/CD | Continuous Integration / Continuous Deployment |
| CIA | Confidentiality, Integrity, Availability |
| EAL | Evaluation Assurance Level |
| EU-CSA | European Cybersecurity Act [5] |
| ISO | International Standards Organization |
| OSS | Open Source Software |
| PE | Process Element |
| SaaS | Software as a Service |
| SDLC | Secure Development Lifecycle |
| SSDF | Secure Software development Framework |
| TAM | Technical Architecture Modeling |
| ToE | Target of Evaluation |

# Chapter 12 Bibliography

[1] Christof Ebert, Gorka Gallardo, Josune Hernantes and Nicolas Serrano: “DevOps”; IEEE Software, Vol.33, No. 3, p. 94-100, 2016

[2] DOD 5200.28-STD "Department of Defense Trusted Computer System Evaluation Criteria", 1985

[3] ECSO WG1 Standardisation, certification, labelling and supply chain management: “STATE OF THE ART SYLLABUS, Overview of existing Cybersecurity standards and certification schemes”; available at https://ecs-org.eu/documents/publications/5a60b8bf83f7c.pdf, retrieved 09/03/2020

[4] European Commission: “„Resilience, Deterrence and Defence: Building strong cybersecurity for the EU“; JOIN(2017)450 final, published 13/09/2017

[5] Regulation (EU) 2019/881 of the European Parliament and of the Council of 17 April 2019 on ENISA (the European Union Agency for Cybersecurity) and on information and communications technology cybersecurity certification and repealing Regulation (EU) No 526/2013 (Cybersecurity Act), PE/86/2018/REV/1

[6] https://en.wikipedia.org/wiki/The_Market_for_Lemons

[7] https://en.wikipedia.org/wiki/Certification, retrieved 09/03/2020

[8] ISO 15408, “Common Criteria for Information Technology Security Evaluation”, Version 3.1, Revision 5, 2017

[9] M. Howard: “Building more secure software with improved development processes; IEEE Security & Privacy, Volume: 2 , Issue: 6 , 2004

[10] ISO/IEC 27034, ““Information technology – Security techniques – Application security”, 2011

[11] https://www.techrepublic.com/blog/it-security/the-cia-triad/, retrieved 23/06/2020

[12] D. Dodson, M. Souppaya, K. Scarfone : “Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF)”, NIST Cybersecurity White Paper, https://doi.org/10.6028/NIST.CSWP.04232020, April 2020

[13] Newhouse W, Keith S, Scribner B, Witte G (2017) National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-181. https://doi.org/10.6028/NIST.SP.800-181

[14] National Institute of Standards and Technology (2018), Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1. (National Institute of Standards and Technology, Gaithersburg, MD). https://doi.org/10.6028/NIST.CSWP.04162018

[15] Migues S, Steven J, Ware M (2019) Building Security in Maturity Model (BSIMM) Version 10. Available at https://www.bsimm.com/download/

[16] BSA (2019) Framework for Secure Software. Available at https://www.bsa.org/reports/bsa-framework-for-secure-software

[17] Hong Fong EK, Wheeler D, Henninger A (2016) State-of-the-Art Resources (SOAR) for Software Vulnerability Detection, Test, and Evaluation 2016. (Institute for Defense Analyses [IDA], Alexandria, VA), IDA Paper P-8005. Available at https://www.ida.org/research-and-publications/publications/all/s/st/stateoftheartresources-soar-for-software-vulnerability-detection-test-and-evaluation-2016

[18] Microsoft (2019) Security Development Lifecycle. Available at https://www.microsoft.com/en-us/sdl

[19] Open Web Application Security Project (2019) OWASP Application Security Verification Standard 4.0. Available at https://github.com/OWASP/ASVS

[20] Open Web Application Security Project (2014) OWASP Testing Guide 4.0. Available at https://www.owasp.org/images/1/19/OTGv4.pdf

[21] Payment Card Industry (PCI) Security Standards Council (2019) Secure Software Lifecycle (Secure SLC) Requirements and Assessment Procedures Version 1.0. Available at https://www.pcisecuritystandards.org/document_library?category=sware_sec#results

[22] Open Web Application Security Project (2017) Software Assurance Maturity Model Version 1.5. Available at https://www.owasp.org/index.php/OWASP_SAMM_Project

[23] Software Assurance Forum for Excellence in Code (2012) Practical Security Stories and Security Tasks for Agile Development Environments. Available at http://www.safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf

[24] Software Assurance Forum for Excellence in Code (2018) Fundamental Practices for Secure Software Development: Essential Elements of a Secure Development Lifecycle Program, Third Edition. Available at https://safecode.org/wpcontent/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Dev elopment_March_2018.pdf

[25] Software Assurance Forum for Excellence in Code (2010) Software Integrity Controls: An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain. Available at http://www.safecode.org/publication/SAFECode_Software_Integrity_Controls0610.pdf

[26] Software Assurance Forum for Excellence in Code (2017) Managing Security Risks Inherent in the Use of Third-Party Components. Available at https://www.safecode.org/wpcontent/uploads/2017/05/SAFECode_TPC_Whitepaper.pdf

[27] Software Assurance Forum for Excellence in Code (2017) Tactical Threat Modeling. Available at https://www.safecode.org/wpcontent/uploads/2017/05/SAFECode_TM_Whitepaper.pdf

[28] Joint Task Force Transformation Initiative (2013) Security and Privacy Controls for Federal Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Revision 4, Includes updates as of January 22, 2015. https://doi.org/10.6028/NIST.SP.800-53r4

[29] Ross R, McEvilley M, Oren J (2016) Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-160, Volume 1, Includes updates as of March 21, 2018. https://doi.org/10.6028/NIST.SP.800-160v1

[30] Holger Mack, Tom Schröer, “Security Midlife Crisis”, SAP Product Security Summit 2019, SAP Internal, 2019

[31] Anupam Datta, Jason Franklin, Deepak Garg, Limin Jia, and Dilsun Kaynar: “On Adversary Models and Compositional Security”, IEEE Security & Privacy, June 2011

[32] Ken Schwaber and Jeff Sutherland: „The Scrum Guide”, https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf, retrieved 30/06/2020

[33] Dariusz Rogowski, Przemysław Nowak: “Pattern Based Support for Site Certification, The Seventh International Conference on Dependability and Complex Systems DepCoS-RELCOMEX, 2012

[34] Lightship Security: “7 Steps to Common Criteria Certification”, https://lightshipsec.com/download/Lightship-7-Steps-to-Common-Criteria-eBook.pdf , retrieved 30/06/2020

[35] SAP's Technical Architecture Modelling (TAM), https://help.sap.com/saphelp_pd1655_oom/helpdata/en/c8/18d7ae6e1b10148b20d97bb6f5a04c/content.htm?no_cache=true, retrieved 02/07/2020

[36] Natalia Shevchenko: "Threat Modeling: 12 Available Methods", CMU SEI blog, https://insights.sei.cmu.edu/sei_blog/2018/12/threat-modeling-12-available-methods.html, retrieved 02/07/2020

[37] Mozilla WebAppSec/Secure Coding Guidelines, https://wiki.mozilla.org/WebAppSec/Secure_Coding_Guidelines, reztrieved 02/07/20

[38] OWASP Secure Coding Practices-Quick Reference Guide, https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/migrated_content, retriueved 02/07/20

[39] SafeCode: Fundamental Practices for Secure Software Development, https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf, retrieved 02/07/20

[40] UC Berkeley Information Security Office: Secure Coding Practice Guidelines, https://security.berkeley.edu/secure-coding-practice-guidelines, retrieves 02/07/20

[41] Synopsys: Secure Coding Guidelines, https://www.synopsys.com/software-integrity/software-security-services/secure-coding-guidelines.html, retrieved 02/07/20

[42] Trupti Shiralkar, Brenda Grove (atsec): Guidelines for Secure Coding, https://www.atsec.cn/downloads/pdf/secure-coding-guidelines.pdf, 2009, retrieved 02/07/20

[43] https://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis, retrieved 02/07/20

[44] SPARTA Deliverable D5.1, "Assessment specifications and roadmap", 2020

[45] OWASP: Vulnerability Scanning Tools, https://owasp.org/www-community/Vulnerability_Scanning_Tools, retrieved 02/07/2020

[46] José Bacelar Almeida, Maria João Frade, Jorge Sousa Pinto, Simão Melo de Sousa: Rigorous Software Development: An Introduction to Program Verification, Springer 2011

[47] German Federal Office for Information Security: Cloud Computing Compliance Criteria Catalogue (C5), 2020, available at https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/CloudComputing/ComplianceControlsCatalogue/2020/C5_2020.pdf?__blob=publicationFile&v=1, retrieved 28/07/20

[48] ANSSI: Prestataires de services d'informatique en nuage (SecNumCloud), 2018, available at https://www.ssi.gouv.fr/uploads/2014/12/secnumcloud_referentiel_v3.1_anssi.pdf, retrieved 28/07/20