Curriculum for

Certified Professional for Software Architecture (CPSA)® *Advanced Level*

Module EMBEDDEDSEC

Embedded Security for Architects

2025.1-RC1-EN-20250625

iSAQB®

International Software Architecture Qualification Board

Table of Contents

Introduction: General information about the iSAQB Advanced Level

What is taught in an Advanced Level module?

The module can be attended independently of a CPSA-F certification.

- The iSAQB Advanced Level offers modular training in three areas of competence with flexibly designable training paths. It takes individual inclinations and priorities into account.
- The certification is done as an assignment. The assessment and oral exam is conducted by experts appointed by the iSAQB.

What can Advanced Level (CPSA-A) graduates do?

CPSA-A graduates can:

- Independently and methodically design medium to large IT systems
- In IT systems of medium to high criticality, assume technical and content-related responsibility
- Conceptualize, design, and document actions to achieve quality requirements and support development teams in the implementation of these actions
- Control and execute architecture-relevant communication in medium to large development teams

Requirements for CPSA-A certification

- Successful training and certification as a Certified Professional for Software Architecture, Foundation Level® (CPSA-F)
- At least three years of full-time professional experience in the IT sector; collaboration on the design and development of at least two different IT systems
  - Exceptions are allowed on application (e.g., collaboration on open source projects)
- Training and further education within the scope of iSAQB Advanced Level training courses with a minimum of 70 credit points from at least three different areas of competence
- Successful completion of the CPSA-A certification exam

CPSA-A®
Certified Professional for Software Architecture
Advanced Level

Essentials

What does the module “EMBEDDEDSEC” convey?

The module EMBEDDEDSEC conveys the methods to participants to design an embedded system architecture that reflects their security goals. Building on the concepts of the foundation level, the module introduces participants to analysis methods to identify protection worthy assets and derive security goals for embedded systems. It presents qualities, design patterns and technologies which help in achieving security goals and familiarizes participants with common attack patterns. Finally techniques to verify and validate security properties are presented. At the end of the module the participants know approaches to design embedded architectures with security as a quality in mind, and are able to identify security risks and select suitable control measures.

Curriculum Structure and Recommended Durations

Content	Recommended minimum duration (minutes)
1. Introduction	120
2. Analysis	240
3. Verification	120
4. Cryptography	90
5. Attacks	120
6. Embedded Design Considerations	120
7. Embedded Security Design Patterns	270
Total	1080 (18h)

Duration, Teaching Method and Further Details

The times stated above are recommendations. The duration of a training course on the EMBEDDEDSEC module should be at least 3 days, but may be longer. Providers may differ in terms of duration, teaching method, type and structure of the exercises, and the detailed course structure. In particular, the curriculum provides no specifications on the nature of the examples and exercises.

Licensed training courses for the EMBEDDEDSEC module contribute the following credit points towards admission to the final Advanced Level certification exam:

Methodical Competence:	20 Points
Technical Competence:	10 Points
Communicative Competence:	0 Points

Prerequisites

Participants **should** have the following prerequisite knowledge:

- Foundations of software architecture, as they are taught in CPSA-F accredited trainings.
- Practical experience with the implementation and design of embedded systems and typical challenges in developing embedded systems.

Knowledge in the following areas may be **helpful** for understanding some concepts:

- Practical experience in designing safety-relevant systems.
- Practical experience in designing security-relevant systems.

Structure of the Curriculum

The individual sections of the curriculum are described according to the following structure:

- **Terms/principles**: Essential core terms of this topic.
- **Teaching/practice time**: Defines the minimum amount of teaching and practice time that must be spent on this topic or its practice in an accredited training course.
- **Learning goals**: Describes the content to be conveyed including its core terms and principles.

This section therefore also outlines the skills to be acquired in corresponding training courses.

Supplementary Information, Terms, Translations

To the extent necessary for understanding the curriculum, we have added definitions of technical terms to the iSAQB glossary and complemented them by references to (translated) literature.

1. Introduction

Duration: 90 min	Practice time: 30 min

1.1. Terms and Principles

Quality Attribute, Security, Secure Development Lifecycle, Security Property

1.2. Learning Goals

LG 1-1: Definition of Security

Participants know a definition of security and understand it as a quality of the system. Participants understand the relationship and trade-offs to other quality attributes.

Definitions can be found for example in ISO/IEC 25010, ISO/SAE 21434 and IEC 62443.

LG 1-2: Security Properties

Participants know the security properties of the universal triad: confidentiality, integrity and availability. In addition they know additional common properties such as authentication, authorization and non-repudiation.

LG 1-3: Security Lifecycles

Participants understand, that security must be considered in the whole product lifecycle and not just for the development phase. Participants understand that maintaining security requires participation and consideration of all stakeholders. Participants know the typical lifecycle phases (conception, development, production, operation and maintenance, and decommission). Participants know at least one example of a security product lifecycle.

Examples of security product lifecycles are presented ISO/SAE 21434, the Microsoft Security Development Lifecycle and NIST’s Secure Software Development Framework.

LG 1-4: Security Regulations and standards

Participants understand the difference between regulations, standards and guidelines. Participants know examples of regulations, standards and guidelines regarding security.

Example of security regulations, standards and guidelines are UN R 155, UN R 156, ISO/SAE 21434, FDA Guidelines, NIST Standards such as SP800, IEC 62443, IEC 80001-5-1, IEC 60601-4-5, ISO 270xx and ETSI EN 303 645.

1.3. References

2. Analysis

Duration: 120 min	Practice time: 120 min

2.1. Terms and Principles

Asset, Attack Path, Attack Tree, Damage Scenario, Feasibility, Impact, Risk, Security Claim, Security Goal, System Context, System Definition, Threat, Threat Modeling

2.2. Learning Goals

LG 2-1: System definition and context

Participants understand the security purpose of the system context. Participants are able to create a system context view, building on their Foundation Level knowledge. The system context provides necessary information for the security analysis of the system:

- Interfaces and their purpose
- Data Flows from and to external systems
- Assets associated with the system

LG 2-2: Asset and damage identification

Participants understand what assets are and know typical examples. Participants are able to identify assets for a given system. Participants are able to identify damage scenarios an attack can cause these assets.

Examples of assets in embedded systems are personally identifiable information, user's health and safety, intellectual property, hardware components, cryptographic material and communication channels.

LG 2-3: Threat Modeling

Participants know approaches to threat modeling (attacker-, asset-, system-centric) and understand their advantages and disadvantages. Participants understand the role threat modeling plays in the product lifecycle phases and that the model changes over time. Participants understand that threat modeling is an interdisciplinary tasks, that benefits from diverse inputs.

LG 2-4: Threat scenario analysis

Participants know approaches to identifying threats (e.g., Misuse Case Diagram, Data Flow Diagram, STRIDE). Participants understand how architectural views can be employed for threat modeling. Participants know typical methods to refine threats with attack paths (e.g., Attack Tress, Vulnerability Analysis, Kill-Chains). Participants are able to identify and analyze threats for a given system.

LG 2-4: Risk Assessment Methods

Participants understand the goal of assessing the risk of threats and the associated damage scenarios. Participants know approaches to classifying and rate attack risks.

Examples for common rating systems are CVSS, ISO/SAE 21434 and MITRE's Medical CVSS rubric.

2.3. References

[Shostack 2024]

3. Verification

Duration: 90 min	Practice time: 30 min

3.1. Terms and Principles

Dynamic Testing, Static Analysis, Verification, Validation

3.2. Learning Goals

LG 3-1: Verification Goals

Participants understand the goals of security testing. Participants know classifications of security testing:

- Dynamic Application Security Testing
- Static Application Security Testing
- Interactive Application Security Testing

Participants understand the advantages and disadvantages of static and dynamic methods.

LG 3-2: Static Analysis

Participants know static analysis techniques such as

- Software Composition Analysis
- Semantic Code Analysis
- Vulnerability Checking
- Taint Analysis.

LG 3-3: Dynamic Testing

Participants know dynamic testing techniques such as

- Fuzzy Testing
- Dynamic Taint Analysis
- Vulnerability Scanning
- Threat Mitigation Testing
- Robustness Testing

LG 3-4: Penetration Testing

Participants understand the goal of penetration testing. Participants understand the relationship of penetration testing to other verification methods. Participants know the steps of a penetration test:

1. Test Scope Definition
2. Target Reconnaissance
3. Vulnerability Analysis
4. Proof of Concept Exploits

5. Vulnerability Report

3.3. References

4. Cryptography

Duration: 90 min	Practice time: 0 min

4.1. Terms and Principles

Asymmetric Cryptography, Entropy, Hashing, Key, Post-Quantum Cryptography, Secret, Symmetric Cryptography, Randomness

4.2. Learning Goals

LG 4-1: Goals of Cryptography

Participants know the goals of cryptography (Authenticity, Confidentiality and Integrity). Participants know how a basic cryptographic function works.

LG 4-2: Symmetric Cryptography

Participants understand what symmetric cryptography is. Participants know use cases, advantages and disadvantages of symmetric cryptography. Participants know examples of recommended symmetric algorithms (e.g. from NIST or BSI).

LG 4-3: Asymmetric Cryptography

Participants understand what asymmetric cryptography is. Participants know use cases, advantages and disadvantages of asymmetric cryptography. Participants know examples of recommended asymmetric algorithms (e.g. from NIST or BSI) and are aware of the need to future-proof applications for post-quantum attacks.

LG 4-4: Secure Hashing

Participants understand what a hash function does. Participants understand quality goals of hash functions (preimage resistance, second-preimage resistance, collision resistance, avalanche effect and uniqueness of output). Participants know use cases of hashing. Participants know examples of recommended hashing algorithms (e.g. from NIST or BSI).

LG 4-5: Key Derivation Functions

Participants understand the use of key derivation functions. Participants understand the difference in qualities compared to hashing functions. Participants know examples of recommended key derivation functions and their uses (e.g. from NIST or BSI)

LG 4-6: Randomness and Entropy

Participants understand why cryptographically-secure random values are an important cornerstone of cryptography. Participants know how random values can be generated through random number generators (RNGs) and pseudo-random random number generators (PRNGs). Participants are aware of common pitfalls, when generating random values.

4.3. References

[Ferguson 2010]

5. Embedded Security Threats

Duration: 90 min	Practice time: 30 min

5.1. Terms and Principles

Attack, Attack Surface, CVE, CWE, Threat Agent, Vulnerability, Weakness

5.2. Learning Goals

LG 5-1: Attacker Motivations and Knowledge

Participants know the levels of capabilities (script kiddy, programmer, security expert, state actor/competitor, etc.) and motivations (fun, research, monetary gain) attackers exhibit. They should understand the issues with attacker-based risk approaches.

LG 5-2: Attack Terminology

Participants understand the difference between weaknesses and vulnerabilities. They understand the concept of an attack surface and how it relates to weaknesses and vulnerabilities.

LG 5-3: Security Information Sources

Participants know sources from which information about attacks, vulnerabilities and weaknesses can be gathered (e.g., CVE and CWE database, OWASP, SANS Institute, CISA, BSI, or UN R 155 for automotive).

LG 5-4: Common Attack Patterns

Participants understand typical weaknesses, attack patterns and their effects. Examples for these are overflows, injections, privilege escalations, denial of service, attacker-in-the-middle, reverse engineering, social engineering. Further examples can be found in the OWASP (IoT) Top 10 and aforementioned security information sources.

LG 5-5: Hardware Attack Surfaces

Participants understand that attacks on embedded systems are not limited to software-based attacks, but can also be conducted via the system’s hardware. Participants know that hardware attacks can be invasive, therefore damaging the device under attack, or non-invasive. Participants know hardware-based attacks such as side channel attacks, voltage glitching, or microprobing.

5.3. References

[CVE-Database], [CVE-Database], [OWASP Top 10], [OWASP IoT Top 10]

6. Embedded Security Design Considerations

Duration: 90 min	Practice time: 30 min

6.1. Terms and Principles

Security by Design

6.2. Learning Goals

LG 6-1: Security as a Quality in Embedded Systems

Participants understand security as a system quality and its relation to other quality requirements. Participants understand security as a quality in the context of ISO 25010 and the relation of these qualities to the threat modeling analysis technique STRIDE. Participants understand that embedded security is a cross-cutting concern, that needs to be addressed at the system, hardware and software level and can not be solved only in software.

Participants understand, that the physical access users can gain to the devices pose an uncircumventable risk, therefore it should be considered that the compromise of single device should not compromise all products (e.g., when the same symmetric key is used in all devices).

LG 6-2: Safety and Security

Participants understand that embedded systems can influence the physical world and pose additional safety risks that need to be addressed.

LG 6-3: Guiding Principles

Participants understand that good software engineering practices help designing a more secure system. They are able to explain why a modular design supports achieving security goals.

In addition, participants know design principles to building secure systems and how they are reflected in embedded systems, examples include:

- Defense in depth (e.g., encrypted firmware with additional hardware readout-protection),
- Least-privilege Principle (e.g., reduced access to peripherals or enforcement of memory regions),
- Data Minimization (e.g., which information to log or transmit),
- Self-protection (e.g., voltage-glitch detection, secure states),
- (No) Security by Obscurity
- Input and Output Validation

LG 6-4: Resource Restrictions

Participants understand that resource constraints limit the solution space for embedded systems. Examples of such limitations are:

- Space, Computation Power, or Energy: The implementability of cryptographic algorithms might be limited by available peripherals, or the speed with which algorithms can be computed. Reliable update strategies, such as AB-Partitioning might not be implementable.

- Networking and Connectivity: The unavailability of an internet connection, a trusted time provider, or other communication paths might hinder updateability, usage of time reliant authentication mechanisms or ensuring secure, auditable logging.
- Memory: Different types of memories are used, which might limit how often they can be overwritten. This can be used as a benefit, e.g., for One-Time Programming Keys, but also cause drawbacks in case these keys are compromised. Enforcement of roll-back protections through hardware fuses might be limited by the amount of available fuses.

LG 6-5: Software Updates

Participants understand the need for software updates and the challenges deploying updates to embedded devices pose. Participants know the security trade-off of updatable devices, as this mechanism increase the devices attack surface. Participants know possible solutions to securely deploy updates to embedded devices (e.g., signed and encrypted firmware packages, secure version numbers)

LG 6-6: Secure Implementation

Participants understand how a well engineered implementation supports the security goals. Participants know standards and guidelines to reduce the likelihood of introducing defects during the implementation such as MISRA, CERT Coding Guidelines or OWASP’s Secure Coding Practices.

6.3. References

[Fernandez-Buglioni 2013], [Schumacher 2006]

7. Embedded Security Design Patterns

Duration: 120 min	Practice time: 150 min

7.1. Terms and Principles

Security by Design

7.2. Learning Goals

LG 7-1: Authentication and Authorization

Participants know methods, patterns and technologies to ensure authentication of entities and manage authorization for actions taken on the system.

Authentication example patterns are

- Symmetric and Asymmetric Authentication (e.g., using Universal Diagnostic Services),
- Credentials,
- Multi-Factor Authentication (biometrics or secure hardware tokens),

Authorization example patterns are

- Access Control Lists,
- the Authenticator Pattern,
- Capabilities,
- Mandatory Access Control,
- Multi-Level Security.

Participants understand that users in embedded systems might be other systems and system-to-system authentication and authorization needs to be addressed, compared to user-to-system authentication. Participants understand that user interaction with the device might be limited, when selecting authentication mechanisms.

LG 7-2: System Integrity

Participants know methods, patterns and technologies to ensure the system’s integrity and protect the system against tampering.

Examples to ensure software integrity are

- immutability,
- run-time only mutability,
- software signing,
- Secure Boot (e.g., using a secure element or a trusted computing base),
- encryption of user data.

Examples to ensure integrity of operations are

- use of the memory protection unit to ensure isolation and memory access control,
- separation of persistent mutable data from immutable, executable data,
- usage of isolation and sandboxing,
- software memory protection (Stack Canaries, address space layout randomization),
- resource exhaustion protections,
- control flow checking (e.g., software based encoding or hardware watchdogs),

LG 7-3: Communication

Participants understand the necessity of ensuring confidentiality, integrity and availability of communication. Participants know mechanisms to harden communication against attacks such as

- message counters to detect dropped messages or availability issues,
- encryption (e.g., with AEAD algorithms),
- segregation of communication networks and
- security protocols such as MACSec, IPSec, TLS,
- secure On-Board Communication for CAN, or
- air gaping critical systems

Participants understand that embedded buses are limited in bandwidth and message size, therefore security mechanisms for protecting communication over such buses might have lower security guarantees compared to TLS or other typical communication protection.

LG 7-4: Hardware Security

Participants understand what role the underlying hardware plays in achieving security goals. Participants know how isolation concept can be realized with hardware support, what a Hardware Security Module or a Trusted Platform Module is. Participants know examples of hardware security mechanisms such as Physical Unclonable Functions (PUF) or Fuses. Participants know that hardware interfaces might need to be disabled for a production deployment, e.g., serial and debug interfaces Participants know examples for anti-temper protection such as temper sensors, coatings and potting, physical access protection.

Participants understand the concept of secure enclaves and now examples of implementations, such as the ARM’s TrustZone, the Confidential Compute Architecture, or the Secure Hardware Extension (SHE).

7.3. References

[Fernandez-Buglioni 2013], [Schumacher 2006]

References

This section contains references that are cited in the curriculum.

C

- [CVE-Database] The MITRE Cooperation: Common Vulnerabilities and Exposures. https://www.cwe.org/
- [CVE-Database] The MITRE Cooperation: Common Weakness Enumeration. https://cwe.mitre.org/

F

- [Ferguson 2010] Ferguson, N., Schneier, B., Kohno, T: Cryptography Engineering: Design Principles and Practical Applications. Wiley, 2010
- [Fernandez-Buglioni 2013] Fernandez-Buglioni, F: Security Patterns in Practice: Designing Secure Architectures Using Software Patterns. Wiley, 2013

O

- [OWASP Top 10] Open Web Application Security Project: Top 10 Web Application Security Risks. https://owasp.org/www-project-top-ten/, 2021
- [OWASP IoT Top 10] Open Web Application Security Project: OWASP Top 10 Internet of Things 2018. https://owasp.org/www-project-internet-of-things/, 2018

S

- [Shostack 2024] Shostack, A.: Threat Modeling Designing for Security. Wiley, 2014
- [Schumacher 2006] Schumacher, M., Fernandez-Buglioni, F., et al.: Security Patterns: Integrating Security and Systems Engineering. Wiley, 2006