logic technology

Strengthen Security

IoT Secure SDLC

V. 1-2023

Secure SDLC

In this whitepaper we will discuss how you can develop and maintain IoT products and services, to best shield them and your customers from security incidents, while conforming to the European Cyber Resilience Act in its current draft status.

Although the software is the biggest portion of an IoT product and its infrastructure and thus the most prone to security issues, the software ‘lives’ within electronics. Without hardware no software and thus no IoT.

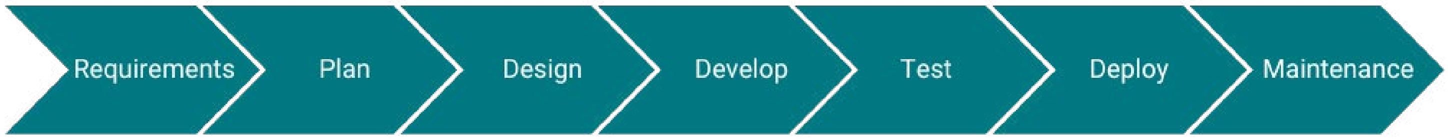
It is during the requirements specification phase, when hardware and software specifications must be carefully orchestrated to ensure maximum overall security. A device that stores critical or private data, or a device that can be updated over-the-air most likely will use a cryptographical algorithm with a private key to ensure data safety, integrity and security. The key therefore may not be exposed. Although one can use an isolated section of non-volatile memory to store this key, the most secure way is to implement a Trusted Platform Module (TPM) in the hardware to create a Root of Trust or Chain of trust between hardware and software.

Therefore, in a well-balanced secure development process, the security requirements for the software are matched with the security requirements of the hardware. From then on, hardware and software development processes branch off and come together again later in the development cycle.

Securing IoT

In this paper, we will focus on the Software Branch of the Life Cycle within the IoT ecosystem. Securing IoT in itself implies that the entire ecosystem needs to be secure. So end-devices, gateways, cloud backend, apps are all involved as independent systems. From a software point of view, all parts in all systems need to be developed in a secure way, from the firmware, RTOS, communication stacks and application in end-devices to cloud services and software implementations, network protocols etc. No doubt that this a complex task especially since the SDLC consists of different phases and in order to maintain maximum security needs a specific security aware implementation in every phase.

In the next sections, we will look at the different SDLC phases and their security considerations.



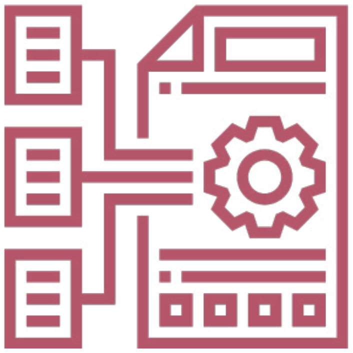
1. Requirements

It is precisely in the requirements phase where security considerations are essential; The requirements are the basis of the specifications for the design, development, maintenance and deployment. Security therefore must be embedded in the requirements throughout.

Besides that, personnel training and establishment of clear and well-defined processes is beneficial in meeting certain cases of requirements for legal or regulatory purposes.

Security considerations

- Password change policies for (end-)users
- Recovery planning
- Update policy
- Cost, benefit and risk analysis
- Third party dependencies
- Security standards
- Certification objectives
- Potential IoT threats
- IoT attack vectors



In respect to security, the software requirements consist out of a part related to the context and a part related to functionality.

Requirements must remain consistent and a mechanism must be in place to regularly check them against on-going modification. There are several techniques available in respect to security such as quality gateways and bug bars. The former is a way of checking the requirements for completeness and correctness whereas the latter puts thresholds on the vulnerabilities.

Threat Modelling
Risk analysis of the assets as well as their internal and external interactions and dependencies, together with threat modelling that charts structural vulnerabilities are other areas of focus. Threat modeling starts in the requirements phase and works through to the design phase.

Given the nature of IoT ecosystems, requirements are no longer static documents. As ecosystems adapt to new capabilities, exchangeability with new and legacy systems can create potential vulnerabilities ruled out at the beginning. It is inevitable that requirements change over time and special care must be taken to regularly iterate them in the light of security.

2. Software Design

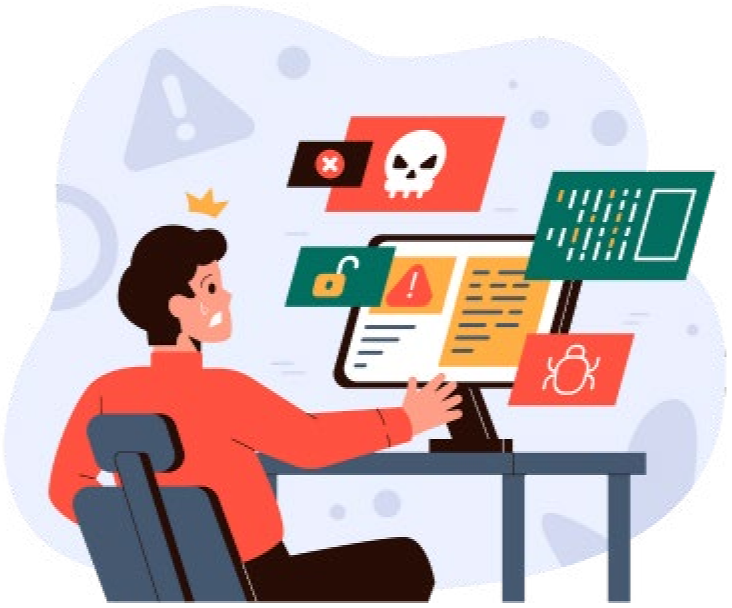
The Software Design phase is the translation of the requirements into a set of documents and drawings with system specifications. Basically they lay-out how the device or application will work.

The functional aspect of the design used to be the primary goal, but today a risk-based approach should be leading to ensure security throughout the design.

The threat model developed in the requirements phase together with an analysis of the areas of potential attack helps in ruling out potential security breaches.

In this phase a proper strategy for a chain of trust, recovery plan and the integration of security mechanisms are crucial for an effective sustain or maintenance cycle once the product is released.

A security Architectural design leans on the CIA triad (Confidentiality, Integrity and Availability), taking into account aspects like access control, policy configuration and security lifecycle. It is this part of the design that is particularly vulnerable to security leaks as it needs to detail the security privileges of all devices, services and applications that have some sort of interaction.



logic technology

3. Development

The classical way of translating specifications to functioning code with the edit-compile-debug stigma no longer works for security centric software development without specific secure coding guidelines or programming standards.

Source code must be structured, readable, transferable, maintainable yet optimized to a certain level and free of vulnerabilities. Development of secure software relies on (industry) coding standards to provide security by design code from the beginning.

In today's development setup with continuous integration practices, the software development process continuous in parallel to testing activities. Because of this dynamic environment, automated tools such as Static code Analysis tools aid in the detection of security issues.

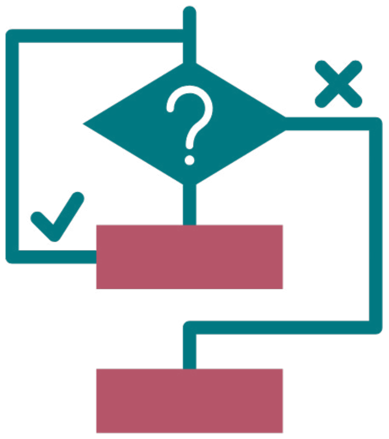
On a higher level Static Application Security Testing (SAST) tools can automate the security processes and eliminate application level vulnerabilities.

Special care must be taken when consuming third party or Open Source components. In many cases these are treated as black boxes, but in a security centric environment, they should be tested and checked for known vulnerabilities before integration in the IoT system.

It may be a no-brainer that the development process must be organized in a way that every new version of the software is less vulnerable than the previous. Version Control Systems, secure bootstrap are some of the methods to aid in achieving these goals.

4. Testing

Where in the past, testing was mostly done on a (sub)system level specifically to validate functional implementation, when we speak in terms of security, new dimensions have entered the equation. When developing test specifications for secure IoT, the risk and threat analysis of the entire IoT ecosystem must be considered, including their interfaces, dependencies and data flows.



Full functional test
It is not impossible, but extremely difficult to perform a full functional system test on an entire IoT ecosystem all at once, so the security testing strategy needs to identify all relevant elements in the system and specify how each element and sub-element can be best tested.

There are many test strategies to choose from but it is important that Testing and Checking is done as early as possible in the SDLC. As an example consider an IoT device based on a main stream Linux kernel any many open source libraries. The kernel version is specified and based on that all necessary

Software Composition Analysis
A software composition analysis tool can be used to check if any of the packages contain known vulnerabilities and offer manual or automated remediation to eliminate them before any further development or testing is done. When coding has started, manual or automated testing with static or dynamic code analysis help to validate functionality against the requirements and check for any unwanted of vulnerable behavior. and peer review.

Peer Source Code Review
As (system) complexity and interdependency grow, a peer source code review, although time and resource consuming, might be a necessary activity to enhance overall system security. Human intervention in testing will remain a decisive factor as long as automated tools continue to generate large numbers of false positives. It is here, where automated testing is preferred because of their efficiency, consistency and inherited testing documentation.

5. Deployment & Integration

The deployment phase in an IoT environment requires a solid bug tracking and change management system to safeguard that any deployment of new firmware or software on any level keeps its operability with the rest of the ecosystem and does not introduce vulnerabilities. In many cases, deployment may not be limited to just end devices, but may also involve back end or third party systems making the deployment steps complicated. Deployment therefore is an activity that needs to be carefully communicated and scheduled both with internal as well as external stakeholders.

There are several documented deployment strategies and depending on the type of application, number of users or installed devices, the age or maturity of the ecosystem, one must carefully select the strategy best suited, and this should be done early in the design phase.

6. Maintenance

Often called the last phase of IoT SDLC, but not the least important one. Once deployed, the software needs maintenance to ensure availability, stability and functionality.

From a security perspective, this is the phase where the entire ecosystem needs to be monitored for threats and disruptions. Key in this phase is the implementation of a vulnerability assessment and penetration test strategy as well as back up and redundancy schemas for operational and security fall-back scenarios.

Other tasks during the maintenance phase include: software update management, regulatory compliance and secure software and device disposal. The latter entails the mechanisms to ensure that all data, access and authentication information is deleted in a secure way when end-point devices, gateways, servers, discs etc are taken out of service.

logic technology

Conclusion & Recommendations

Security is a fundamental principle across all phases in the IoT SDLC. It is crucial to have a good documentation management system that supports the SDLC process for traceability, monitoring and auditing.

Documentation that finds its origin in well defined security requirements carried forward through the IoT Design, Implementation, Testing, Deployment and finally the Maintenance phase.

Each phase has its own set of tools, processes and procedures in aid of achieving the security requirements. Setting up and maintaining a suitable Secure SDLC in itself is very complex.

Our recommendations are to:
• Make use of tools wherever possible to ease document generation and overall traceability
• Start implementing security measures early in SDLC
• Adhere to common industry standards for software coding
• Automate repetitive processes such as testing for maximum consistency
• Use special software components for secure and reliable device data management

There's always **a Logic solution**

Tools

These tools are our recommendation when you want to strengthen security within your company.

Keep your company and your customers safe with tools for security by design, certification & testing.

Certification & Traceability
Managing software requirements using informal business tools like spreadsheets might work for a while, but even for smaller projects and project teams, managing the impact of changes throughout the entire life cycle can be difficult.

LDRA has been a trusted supplier of Logic Technology for over 10 years. Their tool suite's open and extensible architecture integrates software life-cycle traceability, static and dynamic analysis, unit test and system-level testing on virtually any host or target platform.

Request a free trial

Application Lifecycle Management
ALM, or Application Lifecycle Management, is designed for managing the entire lifecycle of an application, from initial design to retirement. From better quality, to a faster time-to-market, ALM provides a comprehensive framework for managing a secure design.

Our supplier Kovair offers state of the art ALM tooling, which integrates with all existing software.

Request a live demo

Secure File Systems | Encryption
File system solutions are carefully engineered to protect critical system and user data from corruption where power loss may occur. Tuxera has been our partner for years with their Reliance Edge, Reliance Nitro and Reliance Sense solutions.

Contact us for your free trial

Open Source Software Composition Analysis
Open source components have become an integral part of today's software development processes. SCA tools automatically detect the open source components in your applications and help you manage the different aspects related to your open source usage. Mend is our partner for open source management. Their solution Mend.io SCA is the gold standard in open source security.

[Read more on managing open source software]

Secure Databases

A database management system provides provides protection and security to a database. In the case of multiple users, it also maintains data consistency.

If you're talking databases, we say there's no one better than McObject. eXtremeDB is an extremely fast and reliable database management system.

[Read more on secure databases]

Trusted Platform Module (TPM)
The use of a TPM is an important first step within your prod-uct design. By using a solution by our partner Insyde, called InsydeH2O, you get acces to a the industry's most trusted UEFI BIOS.

[Discover Insyde H2O]

There's always **a Logic solution**

Wondering how to integrate security measures and tooling within your product or company?

Schedule a call with a product matter expert to have 30 years of product development knowledge by your side.

[Let's get started!]

logic technology