Study of the cybersecurity aspects of the RED Directive

Thierry Didi,
May 2025
Tidiwi

In a previous article I was interested in the security of connected objects, as promoted by the European community or by non-profit organizations.

Following an exchange with an industry and reading a post on LinkedIn, I noticed that the new cybersecurity requirements introduced by the RED Directive are still unknown to some players in the sector. Their impact on the marketing of products often seems to be poorly assessed: some minimize the consequences, thinking that a simple visit to the laboratory will suffice, while others anticipate a complete redesign of their equipment. However, the deadline is fast approaching, with an entry into force scheduled for August 2025.

Therefore, I have decided to analyse the new RED directive and the associated standards in a neutral way.

This new version introduces three new articles:

- 3.3§d: "*radio equipment does not harm the network or its functioning nor misuse network resources, thereby causing an unacceptable degradation of service*"
- 3.3§e: "*Radio equipment incorporates safeguards to ensure that the personal data and privacy of the user and of the subscriber are protected*"
- 3.3§f: "*Radio Equipment supports certain features ensuring protection from fraud*"

For each of these three articles, I have chosen to address the following topics:
- Which products are affected,
- What new constraints are applicable to these products,
- What actions must be implemented by manufacturers to comply with them.

This document is not intended to replace the standards associated with the RED Directive. Rather, its objective is to highlight the types of threats that equipment will have to guard against in order to meet the requirements of this directive. It is then up to each manufacturer or compliance officer to:
- Identify the applicable standards,
- To determine whether the equipment falls within the scope of the new articles concerned,
- Implement, if necessary, the required functionalities,
- And to compile the appropriate technical documentation, in order to demonstrate the conformity of the product.

This document is based on a personal study of European directives and associated standards. It therefore has no legal value and is based solely on my understanding of the texts and my knowledge of the embedded systems. If you have any comments or suggestions for improvement, do not hesitate to let me know by email: thierry.didi[at]tidiwi[dot]com.

Introduction

This document is only interested in the three new articles that deal with safety, and not on compliance with the RED directive in general.

It is important to note that there is no measurement tool that can automatically verify a product's compliance with the requirements of the cybersecurity standards associated with the RED Directive. Compliance is based above all on the analysis carried out by the product designers, who will have to document in the RED technical file the way in which each constraint has been taken into account and satisfied. The use of a notified body does not therefore exempt from this assessment, which can only be carried out by the teams in charge of design.

It is also important to note that a manufacturer can carry out a self-certification procedure[1] for its product if it has integrated into its product (and documented) the constraints indicated in the harmonized standards:

> *'Radio equipment, which is in conformity with harmonised standards or parts, thereof the references of which have been published in the Official Journal of the European Union, shall be presumed to be in conformity with the essential requirements set out in Article 3 covered by those standards or parts thereof.'* [2]

Before we get into the details of constraints and solutions, it's important to define some of the terms used in this document and in the specifications:

- **Security Asset**
  A sensitive or confidential setting or feature that ensures the confidentiality of *a user's* personal data, a telecommunications *network asset*, or a *financial asset*.
  For example, a user's GPS location is personal *data* that is a *Privacy Asset* – but a feature that stores this data is also a *security asset*. The code for this function is therefore one of the elements to be protected, as is the position data itself.

[1] ANNEX II of the RED Directive
[2] Article 16 of the ADR Directive

- **Privacy Asset**
  Allows the direct or indirect identification of a *user* (natural person) – e.g. an IP address, a username, an identifier, a MAC address, etc. The protection of privacy-related assets is addressed in Article 3.3(e) of the RED Directive. These assets include:
  - **Personal data**[3]
    Any information relating to an identified or identifiable natural person (hereinafter referred to as the "data subject"); an 'identifiable natural person' is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;

  - **Traffic data**[4]: all data processed for the purpose of routing a communication over an electronic communications network or for invoicing it.

  - **Location data**[5]: all data processed in an electronic communications network indicating the geographical position of the terminal equipment of a *user* (natural person) of a publicly available electronic communications service.

- **Financial Asset**
  Any resource related to the management of monetary value. The protection of financial assets is dealt with in Article 3.3(f) of the RED Directive.

- **Network Asset**
  Any network-related resource whose compromise could disrupt the operation of the network (communication protocols, wired Ethernet interfaces, WIFI, Bluetooth, remote management interface (SSH, etc.), SSH or WIFI passwords, etc.). The protection of network assets is addressed in Article 3.3(d) of the RED Directive.

- **RED Technical File:** To prove the compliance of a device with the RED Directive, the manufacturer must draw up a technical file. The minimum content of this technical file is described in Annex V of the RED Directive.

- **Factory Status**: The state of an asset when it leaves the factory, or when it is reset to a default state.

- **User**[6]: any natural person using an electronic communications service accessible to the public for private or professional purposes without necessarily being subscribed to this service.

- **Known Vulnerabilty**: There are several databases that describe publicly known vulnerabilities in open source or commercial software. These databases contain a description of the security vulnerability, possible effects, and possible fixes.

[3] Article 4 of European Regulation (EU) 2016/679
[4] Article 2(b) of Regulation (EU) 2002/58/EC
[5] Article 2(c) of Regulation (EU) 2002/58/EC
[6] Article 2(a) of European Regulation (EU) 2002/58/EC

The new RED Directive

As mentioned in another article dedicated to the security of connected objects[7], there are three harmonized standards that guarantee compliance with the RED directive. These standards are:

- EN 18031-1:2024: Common Security Requirements for Internet Connected Radio Equipment. Used to verify compliance with RED 3.3-§d
- EN 18031-2:2024: Common Security Requirements Equipment Processing Personal Data: Used to verify compliance with RED 3.3-§e
- EN 18031-3:2024: Common Security Requirements for Equipment Processing Virtual Money or monetary value: Used to verify compliance with RED 3.3-§f

Each of these standards may or may not bring new constraints on a product, whether in terms of electronics or software.

This Directive does not apply to medical equipment or in vitro diagnostic devices, automotive equipment, equipment intended for civil aviation, electronic road toll charging, which are covered by other Directives[8].

The following sections discuss these constraints in the context of network protection, privacy and fraud protection. For reasons of personal interest, I was primarily interested in the protection of personal data, and the sections are therefore listed in this order:

- Protection of personal data.
- Network protection.
- Fraud protection.

[7] https://tidiwi.com/shared-files/3944/?Technical%20Paper%20Securit%C3%A9-V1-fr.pdf
[8] Point (15) and (16) of COMMISSION DELEGATED REGULATION (EU) 2022/30

Article 3.3§e: Protection of personal data

Devices concerned by Article 3.3§e

The EN 18031-2:2024 specification introduces constraints related to cybersecurity. These constraints are intended to protect the *security assets and privacy assets* embedded in the device. These constraints only apply if the device handles *personal data, traffic data, location data* (i.e. associated with a *user,* a natural person). In this case, these constraints apply:

- *If the device has an internet connection*
- *Or if the device is a toy or equipment intended exclusively for the monitoring of children,*
- *Or if the device is a wearable connected accessory (smartwatch, body-worn sensor, etc.)*

In other words, if the device cannot be associated — either directly or indirectly — with a natural person, then the security asset and privacy requirements do not apply.

However, the notion of association with a user must be interpreted broadly: As soon as the device embeds an identifier (e.g. a serial number, a MAC address or an IP address) that can be linked to a natural person via a remote platform (such as a cloud service), the requirements of this specification become applicable.

Of course, all these facts will have to be justified in the *Technical File*[9] which attests to the compliance of the equipment with the RED Directive.

The constraints introduced by Article 3.3§e

As a first step, the manufacturer must list all the *security assets* and *privacy-related assets* to be protected. It must also list all possible access, legitimate or illegitimate, to these assets.

Then, for each of these security assets, it will have to guarantee (and document and justify) how it meets the following constraints:

- **Access control**: This is to protect read or write access to *privacy* assets and *security* assets. This protection applies of course to legitimate users, but also to malicious users who might try to obtain this data by means of the radio interfaces (Bluetooth, Wifi, Cellular...) that are integrated into the equipment. On the other hand, if the environment in which the equipment is used itself has "physical" or "logical" access control measures, it is not necessary to implement this type of measure in the equipment itself. These measures should be documented in the *RED Technical File*.
  Toys and devices for child tracking incorporate other needs in terms of access control for parents or caregivers.

- **Authentication**: For each type of access control identified, this involves requesting authentication before allowing an external entity to read or manipulate *security assets* or *privacy-related assets*. Different authentication modes will be implemented on user interfaces (e.g. username/password, biometrics, pin code...) and on network interfaces (TLS...). This constraint does not apply if the environment in which the equipment is used has access control measures, or if access to certain personal information, read-only, is part of the normal use of the equipment (e.g., displaying the user's name on the device's screen, or reading a public key from an asymmetric encryption algorithm).

[9] The content of the technical document is specified in Annex V of the RED Directive

If "factory" passwords are set, they must be unique for each device or ask to be changed the first time they are used, possibly with a blank password. Similar constraints apply to other passwords that protect *security assets* or *privacy-related assets*.

Depending on the type of personal data handled by the equipment, this specification may require two-factor authentication.

The specification also requires that authentication mechanisms have a minimum robustness. If password-based, passwords should have some complexity. If they are certificate-based, the device must be able to verify the validity of the certificates through signature and a chain of trust for example.

In addition, the device must be protected to prevent a malicious actor who records an access control communication from then being able to replay it to gain access to a protected resource ("Replay Attack").

Finally, the device must protect itself against brute force attacks, where a malicious user has an infinite amount of time to test a large number of passwords.

- **Secure updates**: If the device contains *security assets* or *privacy-related assets*, it must implement at least one method to update the software, or at least the software components, that manipulate this data, unless that software is stored in ROM (Read Only Memory) or other measures protect these assets for the life of the product. Like what
    - For very basic systems such as sensors, one of these measures may be the replacement of the product.
    - The equipment can be part of a system that protects these assets from vulnerabilities.

  The integrity of downloaded software must be verified, i.e. it must not be possible to install software that does not come from an authorized source.

  The equipment must be able to update automatically, or as a result of user action.

- **Secure Storage**: If the device permanently stores (i.e., in non-volatile memory) *security assets* or *privacy-related assets*, this data must be stored securely (typically via encryption algorithms) and protected from modification (typically via signatures). The protection mode to implement depends on the type of data to be stored and how long it is stored.

- **Communications security**: If the device exchanges *security assets* or *privacy-related assets* with other devices, it must use secure communication protocols. These protocols typically provide data confidentiality, data integrity verification, and protection against replay attacks. These protocols must comply with good security practices.

- **Trace mechanism**: If the device manages *security assets* or *privacy assets*, it must set up a log system to trace all manipulations of this data. Typically, the events to be traced are: creation, modification, destruction of *security assets*, or unsuccessful access attempts ("Access Denied").

The generated events must be time-stamped (with an absolute or relative timestamp depending on the device's capabilities) and then stored in non-volatile memory, unless they are stored outside the device (in the cloud for example).

- **Ability to delete data**: The device must provide a procedure that allows all *personal data to be destroyed*. This procedure will be used in particular in the event of the withdrawal of the product from the market, or in the event of the sale of the product for example.

- **User notification**: The device must implement a mechanism to notify the user when security *assets* or *privacy-related assets* change. This constraint does not apply if the user is notified by mechanisms outside the device (by email or SMS for example).

- **Encryption key management**: Unless there is a very specific justification, the encryption keys used to protect data or communications must have a minimum length of 112 bits, except in specific cases that must be documented.

  Encryption keys that are pre-installed in devices must be unique for each device. It should not be possible to easily derive these keys from the serial number or other device data. This uniqueness is not required:
  - If these keys are used to establish an initial relationship of trust and under controlled conditions,
  - If the functionality associated with the key requires that these keys be common (e.g. for a firmware update).

- **Device Functionality**: The equipment must not incorporate a *known security vulnerability* that would compromise *security assets* or *privacy-related assets*, unless:
  - Following a risk analysis, the consequences of these vulnerabilities have been accepted,
  - A technical solution has made it possible to reduce the risk to a residual risk

  To do this, the manufacturer must maintain a list of software modules and hardware components that are built into their product and that are used to manage *security assets* or *privacy-related assets*.

  In addition, the specification recommends limiting the network features and interfaces available in *factory out-of-work* mode to the bare minimum required for the first commissioning of the device.

  To limit the attack surface of the device, all interfaces that are not useful for the operation of the device must be inhibited (typically the JTAG interface, TCP/UDP ports not used, serial ports not used...).

  All input data used to manipulate *security assets* or *privacy-related assets* must be verified. Typically, the goal is to protect the device from code injection attacks[10], where an attacker could force the device to execute malicious code embedded in a request.

[10] https://owasp.org/www-community/attacks/Code_Injection

- **Cryptography**: Cryptography algorithms used to protect data at rest or in transit should follow best practices.

Article 3.3§d – not disrupting network resources

The EN 18031-1:2024 specification introduces constraints related to cybersecurity. These constraints are intended to protect the *security assets and network* assets embedded in the device.

Devices concerned by Article 3.3§d

This Article applies to any radio equipment capable of communicating by itself over the Internet, whether it communicates directly or through other equipment, [11] i.e. such equipment connected to the Internet executes the protocols necessary to exchange data with the Internet, either directly or through intermediate equipment.

It therefore obviously applies to devices with a cellular network access interface (2G, 3G, 4G, 5G, etc.), equipment with a WIFI interface that connects to the internet via a WIFI access point, for example.

In addition, it applies to all communication interfaces of this equipment, whether these interfaces are radio or wired interfaces[12].

A device that is not intended to connect to the internet (for example a garage door opening system) is therefore not concerned.

There remains an ambiguity, in my opinion, concerning the applicability of Article 3.3§d to equipment:

- Who connect via Bluetooth to a mobile application (a heart rate monitor used for sports for example),
- Or that connect via a radio interface (e.g. Zigbee, or a proprietary radio protocol) to a gateway that is itself connected to the internet (a door opening sensor in a home alarm system for example).
- Or connect to a public or private Lora network.

According to my research on the internet, these devices would be concerned by this article since they connect "indirectly" to the internet. But this information often comes from the websites of consulting firms and not from official websites of the European Community.

But my personal reading of the text of the directive leads me to think that this is not the case since these devices do not implement the protocols necessary to exchange on the Internet (i.e. TCP/UDP/IP). So there is no reason why they should be able to disrupt the network.

For this type of equipment, it will therefore be necessary to deepen this question with legal experts, or with the European Commission, to find out whether Article 3.3§d is applicable or not.

The constraints introduced by Article 3.3§d

If the device falls within the scope of Section 3.3§d, the constraints associated with the protection of *privacy-related assets can easily be transposed* to the protection of *network assets*.

In this case, the affected assets are network *assets* rather than *privacy-related assets*.

- **Access control**: transposition of the access control of Article 3.3§e

- **Authentication**: transposition of Article 3.3§e

[11] Article 1 of COMMISSION DELEGATED REGULATION (EU) 2022/30
[12] Item (8) of COMMISSION DELEGATED REGULATION (EU) 2022/30

- **Secure updates**: transposition of Article 3.3§e

- **Secure Storage**: transposition of Article 3.3§e

- **Security of communications**: transposition of Article 3.3§e

- **Management of encryption keys**: transposition of Article 3.3§e

- **Equipment** functionalities: transposition of Article 3.3§e

- **Cryptography**: transposition of Article 3.3§e

- **Resiliency**: The equipment must protect itself to limit the effects of a denial of service attack on its interfaces in contact with the network. For example, it can filter packets at the entrance, or temporarily inhibit the attacked interface in case of suspected attack...

- **Network monitoring:** If the device is itself a gateway to route traffic from other devices to the internet, it must implement network monitoring functions to detect denial of service attempts that may be coming from one of its interfaces.

- **Traffic control:** If the device itself is a gateway to route traffic from other devices to the Internet, it must implement traffic analysis functions from other devices to detect whether a device is generating suspicious traffic that may disrupt the network. If a suspicious type of traffic is detected, the equipment must take measures to protect the network (blocking certain IP addresses, certain ports, etc.).

Article 3.3§f – Protection against fraud

Appliances concerned by Article 3.3§f

This Article only applies to equipment that allows the holder or user to transfer money, monetary value or virtual currency[13] as defined in Article 2(d) of Directive (EU) 2019/713 of the European Parliament and of the Council[14].

This article therefore applies only to a very limited number of devices (typically payment terminals, etc.).

The constraints introduced by Article 3.3§f

If the device falls within the scope of Article 3.3§f, the constraints associated with the protection of privacy-related assets can easily be transposed to the protection of *financial assets*.

In this case, the affected assets are network assets rather than privacy-related assets.

- **Access control**: transposition of Article 3.3§e
- **Authentication**: transposition of Article 3.3§e
- **Secure updates**: transposition of Article 3.3§e
- **Secure Storage**: transposition of Article 3.3§e
- **Security of communications**: transposition of Article 3.3§e
- **Trace mechanism**: transposition of Article 3.3§e
- **Management of encryption keys**: transposition of Article 3.3§e. In addition, equipment that handles financial data must implement secure booting, to ensure that the firmware that is running is genuine firmware.
- **Equipment** functionalities: transposition of Article 3.3§e
- **Cryptography**: transposition of Article 3.3§e

[13] 'virtual currency' means a digital representation of value that is neither issued or guaranteed by a central bank or a public authority, nor necessarily attached to a legally established currency and which does not have the legal status of a currency or money, but which is accepted as a medium of exchange by natural or legal persons and can be transferred, stored and exchanged electronically;
[14] Item (14) of COMMISSION DELEGATED REGULATION (EU) 2022/30

Techniques to be implemented

Depending on the constraints to be respected, the techniques to be implemented to comply with the RED directive are as follows:

Cryptography

Cryptography is used in the areas of access control, authentication, secure data storage, and secure updates.

In terms of security, it is always recommended to use open source libraries that have been validated by a large number of users, and whose flaws are quickly detected and corrected. On the other hand, it is strongly advised not to implement the cryptography algorithms yourself, because it is almost certain that the implementation carried out will contain flaws.

In the field of embedded software, the mbedTLS library[15] developed by ARM is widely used because it is suitable for processors that do not have very large memory capacities. It implements most modern symmetric (including AES, DES, and 3DES) or asymmetric (including RSA, DH, ECC) cryptography algorithms. It can therefore be used to calculate or verify signatures using several algorithms, to manipulate X.509 certificates, or to perform data encryption/decryption operations. It also implements the TLS/DTLS protocols used in secure communication protocols. This library therefore makes it possible to manage the confidentiality of data, the verification of the authenticity of a third party, the verification of the integrity of a message or a file.

There are of course other libraries such as openSSL, an open source library that is rather used in the Linux world, or wolfSSL which is associated with a commercial license.

Encryption key management

Even if very modern cryptography algorithms are used, they will be completely inefficient if the encryption keys used are easily accessible. The devices use symmetric keys to encrypt data, and asymmetric algorithms to calculate or verify signatures, or to execute secure communication protocols such as TLS or DTLS.

Encryption keys must therefore be protected. There are several ways to do this depending on the possibilities on the electronic board:

- Storage of the keys in the processor's internal flash
  The most basic method of protecting encryption keys is to store these keys in the processor's internal flash memory, as long as this flash memory is not accessible from the outside (so in particular, JTAG will have to be disabled). The firmware images should also not be stored in the firmware image but in a dedicated area of the memory because the firmware image could be easily accessible, especially during an upgrade. This method is not the most secure, but it may be enough to guarantee the confidentiality of the keys.

- External flash key storage
- If the processor does not have an internal flash, the keys will necessarily be stored in external flash ... They will necessarily have to be encrypted themselves with a key generated by the

[15] https://github.com/Mbed-TLS/mbedtls

firmware. The security will then be quite minimalist, but will be sufficient if it is properly justified in the technical document.

- Les MPUs[16]
  Most recent microcontrollers based on ARM Cortex-M or ARM Cortex-A architectures, which are widely used in the Internet of Things (IOT), incorporate an MPU. It is a "module" built into the processor that helps protect parts of the memory so that they can only be accessed by code that runs in privileged mode. If the encryption algorithms run in privileged mode while the application code (including the real-time OS) runs in non-privileged mode, the keys will be protected.

- ARM Trustzone™
  Some ARM Cortex-M processors (and also Cortex-A) have a feature called ARM Trustzone built-in. This feature allows a firmware to be completely separated into two separate firmwares, as if the processor itself were composed of two processors with their own interrupt vectors and memory. This feature allows you to create two "execution environments" named SPE (Secure Processing Environment) and NSPE (Non-Secure Processing Environment). The encryption algorithms will therefore run in SPE, while the rest of the code will run in NSPE. Encryption keys will of course only be accessible to code that runs in SPE.

- The Secure Element
  This is the most effective option, but also the most complex and costly. It consists of equipping the electronic board with a specialized component that will be responsible for storing the encryption keys and carrying out the cryptography operations. These components are designed to withstand the most complex attacks such as side channel attacks. Once the keys are configured at the factory, they cannot be reread by software. However, it is possible to update them from the device's firmware. For example, Microchip's ATECC608B components, or STSAFE-A110 from ST Microelectronics can be used. There are of course other references from integrated circuit suppliers.

- Some SOCs[17] include a Secure Element
  SOCs are widely used in connected objects because they include the processor, the radio part, and also integrate memory. The most recent SOCs incorporate a Secure Element, i.e. a cryptographic processor and protected memory to store encryption keys. This solution is even more efficient than the previous one since it is impossible to even spy on the communications between the processor and the Secure Element (even if the Secure Elements are able to encrypt communications on I2C interfaces). This is particularly the case for some Silicon Labs SOCs, which uses the term Secure Vault™ to refer to this feature.

Secure data storage

All data from *privacy-related assets*, *network assets*, *financial assets* that are stored in permanent memory must be protected.

[16] Memory Protection Unit
[17] System On Chip

- Ideally, this data should be stored in the processor's internal flash, if the processor has it, to prevent this data from being spied on on the processor's data bus at the time it is used.

- The minimum protection to be implemented is to encrypt this data, usually with a symmetric encryption algorithm (usually AES). This protection is not enough to ensure that the data has not been modified since it was stored. Therefore, the signature of the data block to be protected (e.g. an ECDSA signature) should also be calculated [18]to ensure the integrity of the data to be protected. We will therefore need a symmetric key for data encryption, and a private key from an asymmetric algorithm to generate the signature.

- A more elaborate implementation proposed by ARM in ARM Trusted Firmware™ is to separate the code that manages the secure data to run in the Secure Processing Engineer (SPE) environment. This implementation assumes that the SOC implements ARM Trustzone, which exists on ARM Cortex M33 or ARM Cortex M55 processors, among others.

Secure communications

Securing communications most often involves the use of TLS (Transport Layer Security) or DTLS (Datagram Transport Layer Security) protocols. These protocols allow a client (the radio equipment) to verify the authenticity of the server to which it connects. Once the authenticity of the server has been verified, the server can optionally verify the identity of the client, but this assumes that a private key has been stored (often at the factory) in the non-volatile memory of the device. Once this phase has been successfully completed, the client and server define symmetric encryption keys that will be used to encrypt the messages they exchange during that session. New keys will be negotiated with each new session.

All IP stacks that come with SOCs implement these protocols, so it's not complicated to use them. However, these protocols are based on the generation and use of X.509 certificates, the management of which will require the implementation of some security-related processes within the company.

Secure firmware updates

All modern SOCs come with a software environment that includes, among other modules, libraries capable of performing secure updates. These libraries can verify the integrity of software, decrypt it (if it has been encrypted before being transmitted), and store it in internal or external memory, in plain text or encrypted according to the developer's choice, and according to the possibilities of the SOC. If firmware images are stored in external flash, they should ideally be encrypted to protect access to any *"security assets*" that may be on them (including the code that manages critical data).

On the other hand, the implementation of software recovery from a server is the responsibility of the developers. To ensure that the software comes from an "official" server, it is essential to authenticate the server that provides the new software to be downloaded. This authentication is usually carried out by the HTTPS protocol: the server provides an X.509 certificate that is verified by the device using a public key that has been provided to it beforehand, often at the time of factory configuration.

Attack surface limitation

It is recommended to disable any unused interfaces. This is in particular the JTAG interface mentioned above because it gives all access to an attacker.

[18] Elliptic curve digital signature algorithm<

Beyond JTAG, UART ports that are often used as a debug console must be inhibited, unless they are required to provide other functionality to the device (such as a configuration console for example): in this case, this interface will need to be protected by user authentication mechanisms, and the data received on this interface will need to be validated before being processed in order for the device to be protected against "malware injection" attacks. code".

Similarly, all unused TCP/UDP ports must be closed.

Vulnerability management

Identifying *known security vulnerabilities* introduced by external libraries in embedded software is simpler to perform than in more complex software. Indeed, embedded software, due to its strong code size constraints, integrates fewer external libraries (open source or not), which would themselves integrate other external libraries, etc ...

However, these vulnerabilities must still be identified. This step involves a list of all external components integrated into the software, then audits of vulnerability databases (e.g. https://nvd.nist.gov, ...) to check if these components include *known security flaws*.

There are also static code analysis tools specifically designed for embedded software, which are widely integrated into continuous integration processes. These tools make it possible to analyze the source code of equipment and identify if it contains *known security flaws*.

However, communication stacks (Wi-Fi, BLE, Zigbee, Thread, etc.) pose a particular problem: they are often provided in binary format by SoC manufacturers, which prevents any direct analysis of the code. In this context, it is necessary to contact the vendor to obtain information on any known vulnerabilities affecting these components.

The STRIDE method is suitable for identifying threats to a system. This method consists of studying 6 categories of threats, as shown in the following table (excerpt from https://en.wikipedia.org/wiki/STRIDE_model):

Threat	Desired property	Threat Definition
Spoofing	Authenticity	Pretending to be something or someone other than yourself
Tampering	Integrity	Modifying something on disk, network, memory, or elsewhere
Repudiation	Non-repudiability	Claiming that you didn't do something or were not responsible; can be honest or false
Information disclosure	Confidentiality	Someone obtaining information they are not authorized to access
Denial of service	Availability	Exhausting resources needed to provide service
Elevation of privilege	Authorization	Allowing someone to do something they are not authorized to do

Once the vulnerabilities have been identified, they will either need to be fixed, bypassed, or decided that these vulnerabilities are acceptable for the product. This decision will be made following a risk analysis, which will characterize each risk based on its impact on the product and its likelihood in the context of the use of the equipment.

Secure Boot

Although the use of a secure boot is only required for devices that manage financial data, its use is widely recommended in all products. This involves checking each time you start the software that is in memory has not been modified since it was downloaded. Ideally, the implementation of a secure boot

requires that a first bootloader be stored in non-modifiable memory (ROM). Indeed, a secure boot would be of no interest if it were possible to modify the bootloader itself...

In general, secure boot implementations include two bootloaders:

- A non-modifiable bootloader stored in ROM (this bootloader is named BL1 in ARM Trusted Firmware™). Since this bootloader is non-modifiable, it contains a bare minimum of instructions as it cannot be patched if a bug is discovered. Its role is to verify the integrity of a second bootloader (named BL2 in ARM Trusted Firmware) and to execute it.
- This second bootloader can integrate more advanced features since it can be updated if a bug is discovered. Its role will essentially be to verify the integrity of the application (or another interactive bootloader called U-Boot in the case of Linux), and then execute it. Eventually, it may implement fallback strategies to an earlier version if it detects that the application (or U-Boot) code has been modified abnormally.

As with firmware updates, SOC vendors all provide examples of secure bootloader implementations. The most recent SOCs include a first BL1 bootloader in ROM, and allow developers to implement their own bootloader (BL2) based on the examples provided.

About TIDIWI

TIDIWI is a consulting company specializing in the design and development of IOT projects. In particular, TIDIWI has a strong experience in wireless communication protocols and in ultra-low power consumption systems.

TIDIWI is involved in consulting, embedded software development, electronic board development, development of development teams, and project management.

For more information, visit Tidiwi's website: www.tidiwi.com

Or contact thierry.didi[at]tidiwi[dot]com

References

- DIRECTIVE 2014/53/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 16 April 2014

- COMMISSION DELEGATED REGULATION (EU) 2022/30 of 29 October 2021

- EN 18031-1:2024: Common Security Requirements for Internet Connected Radio Equipment.

- EN 18031-2:2024: Common Security Requirements Equipment Processing Personal Data

- EN 18031-3:2024: Common Security Requirements for Equipment Processing Virtual Money or monetary value

- ARM Trusted Firmware-A: https://github.com/ARM-software/arm-trusted-firmware

- ARM Trusted Firmware-M: https://github.com/TrustedFirmware-M/trusted-firmware-m