Cross-platform **O**pen **S**ecurity **S**tack for **Con**nected Device

D1.5 Requirements Elicitation Final Technical Specification

Document Identification			
Status	Final	**Due Date**	29/02/2024
Version	1.0	**Submission Date**	27/02/2024

Related WP	WP1	**Document Reference**	D1.5
Related Deliverable(s)	D1.2	**Dissemination Level (*)**	PU
Lead Participant	BIOT	**Lead Author**	Ainara García
Contributors	3MDEB, CYSEC, TUD	**Reviewers**	Yannick Roelvink, Malvina Catalano Gonzaga, CYSEC
			Rafał Kochanowski, 3MDEB

Keywords:

Device Classification, Security Capabilities, Security Guarantees, Relevant Standard Analysis, Threat Analysis, Use Case Requirements, Functional Requirements, Security Requirements, Interoperability Requirements, Performance Requirements, Usability Requirements, WP Requirements.

Document Information

List of Contributors	
Name	Partner
David Purón	BIOT
Ainara García	BIOT
Yannick Roelvink	CYSEC
Malvina Catalano Gonzaga	CYSEC
Shaza Zeitouni	TUD
Rafał Kochanowski	3MDEB

Document History			
Version	Date	Change editors	Changes
0.1	15/05/2023	Ainara García (BIOT)	Proposed methodology for reviewing requirements.
0.2	04/07/2023	Ainara García (BIOT), David Purón (BIOT)	Changes after first requirement review meeting.
0.3	28/09/2023	Ainara García (BIOT), David Purón (BIOT)	Changes after second requirement review meeting.
0.4	02/11/2023	Ainara García (BIOT), David Purón (BIOT)	Changes after third requirement review meeting. Added important feedback from GA meeting at UMINHO.
0.5	15/01/2024	Ainara García (BIOT), David Purón (BIOT)	Changes after last requirement review meeting.
0.6	09/02/2024	Ainara García (BIOT), David Purón (BIOT).	Additional contribution during Validation Criteria meeting.
0.7	20/02/2024	Malvina Catalano Gonzaga (CYSEC), Ainara García (BIOT)	Merge after reviewers' contributions
0.8	26/02/2024	Ainara García (BIOT)	Final version for QA
0.9	27/02/2024	Juan Alonso (ATOS)	Quality Assurance
1.0	27/02/2024	Hristo Koshutanski (ATOS)	Final version submitted

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	David Purón (BIOT)	26/02/2024
Quality manager	Juan Alonso (ATOS)	27/02/2024
Project Coordinator	Hristo Koshutanski (ATOS)	27/02/2024

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
AM	Authentication and Authorization Mechanisms
AR	Attestation Result
CD	Commissioning Data
CK	Cryptographic Keys
CPU	Central Processing Unit
D1.2	Deliverable number 2 belonging to WP1
D2D	Device-to-Device
DM	Device Management
DMS	Device Management Server
DoS	Denial of Service
EC	European Commission
ENISA	European Union Agency for Cybersecurity
FUP	Firmware Update Package
MFA	Multi Factor Authentication
MMU	Memory Management Unit
MPU	Memory Protection Unit
NI	Network Infrastructure
ODM	Original Device Manufacturer
OEM	Original Equipment Manufacturer
OLC	Operation Logs and Configuration
OTA	Over-the-Air
PUF	Physically Unclonable Functions
TPA	Trust Provisioning Authority
TPM	Trusted Platform Module
TRNG	True Random Number Generators
UAV	Unmanned Aerial Vehicle
UC	Use Case
UFL	UAV Fleet List
VMM	Virtual Machine Monitor
WP	Work Package
FPGA	Field Programmable Gate Array

Executive Summary

The document describes the final results of the CROSSCON project's requirements elicitation activities. In the previous requirements elicitation, D1.2 [29], it was included the development of a new IoT device classification scheme that has been validated by every partner involved in the project, an examination of IoT security relevant standards and best practices, a threat analysis of CROSSCON use cases 1, 2, 3 and 4, and the initial technical specification of CROSSCON requirements. This new document validates the previous developments and includes the threat analysis of a new use case, use case 5, for Secure FPGA Provisioning. Finally, a methodology for the validation of requirements has been applied, taking into consideration new requirements for the WP2, 3 and 4 tasks, and identifying the responsibilities and implementation partners for each functionality, security feature or novelty development to be done during the future stages of the CROSSCON project.

The final version of the requirements elicitation includes functional, security, interoperability, usability, and performance requirements. Firstly, some requirements have been derived from D1.4[29] use cases. To match these requirements to the CROSSCON stack design, secondly other specific WP requirements have been added. These WP requirements have been mapped to the use cases to ensure the work meet user needs and project challenges. The use case partners (3MDEB, BIOT, CYSEC, TUD) are responsible for the implementation and validation of their use case related requirements, while the WP partners (UNITN, UNMINHO, BEYOND, TUD, UWU) are responsible for theirs. The other functional, security, interoperability, usability, and performance requirements have different implementation partners and will be further discussed during the Validation Criteria D1.6.

It is important to highlight that the requirements contained in this document might evolve into lower-level and more detailed technical specification requirements directly targeting the CROSSCON stack as the project progresses.

1 Introduction

1.1 Purpose of the document

The purpose of this document focuses on addressing the requirements for the CROSSCON stack and the demonstrators that will be done in the form of UCs.

This document, after having identified the five use cases (UCs) that the CROSSCON stack will face, aims to lay the foundations of security best practices, and focuses on the requirements that defines the roadmap that the CROSSCON stack should cover to address the identified UCs.

The expected outcome of the document is to gather a range of functional, security, performance, interoperability, and usability criteria that are relevant to the various UCs and aligned with the specific classes of devices covered in the document.

It also intends to establish a shared terminology among all consortium members, promoting a mutual comprehension and facilitating collaborative work.

This is the final version of the document completed at M16 of CROSSCON project, and therefore the limitation sets on the initial version have been already treated. These limitations were:

- Device classification has been already validated, finding the correct balance between strictness and flexibility that makes it helpful for future usage.
- Requirements are elicited from UCs, and therefore have a “Security Services” orientation, rather than a “Security Stack” orientation. To avoid this, WP 2, 3 and 4 requirements have been established, having the “Security Stack” orientation expected and mapping to the use cases security service's needs. As well, the discussions for implementation partners and validation criteria have been important for the understanding of the CROSSCON stack expected design.
- The new Use Case 5 has been already described and some requirements have derivate from it.

1.2 Relation to other project work

The first versions of this document are based on D1.4[29] on the Use Case Definition to map the good practices and requirements identified.

Subsequently, this document will serve as a basis for future work, including the following deliverables:

- D1.6 Validation Criteria
- D2.1 Open Specification
- D3.1 Open Security Stack

Finally, the major impact that this study and document will have is on the definition of the testing and validation criteria for the designed hardware and software components for the CROSSCON stack. The associated deliverable is D5.3 Security Testing and Validation Results of the CROSSCON Stack in Use Cases.

1.3 Structure of the document

This document is structured in 7 major chapters.

Chapter 1 is this introduction, and it aims to prepare the reader to understand the scope of the document.

Chapter 2 presents the Device Classification. This section aims to have an IoT device classification or taxonomy that is important to ensure there is a common alignment within a project like CROSSCON. It is presented in this deliverable because there can be a good matching and better understanding on requirements, Use Cases and the Device expected to cover those requirements.

Chapter 3 presents an overview of an analysis made on relevant standards, industry alliances and some regulations that can be taken into account for the CROSSCON stack requirements and design.

Chapter 4 considers the methodology used for the threat analysis and identifies the assets, threats and mitigation techniques for each of the Use Cases.

With all these inputs and formal analysis, the document presents on **Chapter 6** the Requirements Elicitation Tables divided into Functional, Security, Interoperability and Usability Requirements, Use Case requirements and WP2, 3 and 4 requirements.

Chapter 5 overviews the methodology that has been used to validate and have a consensus on the requirements, including both "Security Stack" and "Security Services" orientation.

Chapter 7 aims to map the requirements to the Use Cases and have a proper space to refer to the necessities of each Use Case in particular.

The document ends with a conclusion presented in **Chapter 8.**

2 Common Terminology

The common terminology that has been approved and applied to Requirement Definition was the following:

- **CROSSCON Stack:** Collection of software elements designed within the CROSSCON project.
- **Device:** Network-connected equipment such as a sensor or gateway which is running the CROSSCON stack. The term Device includes hardware and software.
- **Device Software (or SW):** All software installed in the Device, including not only the CROSSCON Stack, but any other software such as bootloader, operating system, or user applications.
- **Device Hardware (or HW):** Physical elements, such as MCU, CPU, memory, and peripherals.
- **Device Identity:** Unique identifier of a given Device.
- **User:** Entity who has the rights to use a given Device.
- **Third Party:** Entity who does not have the rights of using a given Device.
- **Server:** External service to the Device that communicates with the Device for specific purposes, such as authentication, remote management, data acquisition, or others.
- **Low-end device:** Low-end devices feature microcontroller-class processors (such as Cortex-M), typically without MMU.
- **High-end device:** High-end devices feature application-class processors (such as Cortex-A), typically with MMU.
- **Critical Requirements:** Critical requirements are indispensable elements that must be fulfilled to ensure the success or the viability of the CROSSCON stack.
- **Major Requirements:** Major requirements are important components or conditions that contribute significantly to the success or effectiveness of the CROSSCON stack. Major requisites should be completed to achieve satisfactory outcomes, but the failure to meet them may not necessarily result in outright failure.
- **Minor Requirements:** Minor requirements are less critical or important. These are additional conditions or components that, while beneficial, are considered optional to complete, especially if resources are available.

3 Device Classification

3.1 Introduction

The IoT concept is fragmented per-se, there are so many types of connectable “Things” in the market that in many cases just speaking about “IoT devices” leads to many misunderstandings, as different types of devices might have different capabilities, usage, context conditions, etc. Therefore, having a IoT device classification or taxonomy is necessary to ensure there is a common alignment within a project like CROSSCON.

As a first approach to make this classification, some RFCs and articles that present efforts to standardise different taxonomies on IoT devices were analysed. Some of these references are the following:

- IETF RFC 7228 [1] classifies constrained devices in three classes with respect to their RAM and Flash size, however this focus in very constrained devices (kB memory sizes) which in general terms has less security requirements that bigger devices, and therefore it is not fully applicable to the CROSSCON project.
- The Article “The Classification of Internet of Things IoT Devices Based on Their Impact on Living Things” [2] contains an interesting classification based on the potential impact of a security problem in the device (confidentiality, availability, or integrity) would have on user lives.
- According to the paper “A Survey on Trend and Classification of Internet of Things Reviews” by B. A. Desai, D. M. Divakaran, I. Nevat, G. W. Peter and M. Gurusamy [3], the authors present the trends and classification of IoT reviews based on 6 research areas, namely, application, architecture, communication, challenges, technology, and security. This paper proposes a feature-ranking framework for IoT device classification. The main conclusions of the paper are as follows:
  - Network traffic-based features can be effective in accurately classifying different types of IoT devices.
  - Other potentially useful features such as contextual information, activity logs, and RF parameters can also be used to improve accuracy of IoT device classification.
  - The proposed framework has been shown to outperform existing methods in terms of classification accuracy.
- The article “A feature-ranking framework for IoT device classification” [4] proposes that the classification of IoT devices is rather subject to first classifying their features. It refers to characteristics associated with costs such as costs for obtaining the data, extracting, and storing features, compute resources to run a model with high dimensional features, etc. In addition, in this work, the selection of functionalities extracted from the IoT network traffic is contemplated, based mainly on the application of machine learning models for cases such as device identification, anomaly detection, and attack detection.
- Although some conclusions may be interesting to CROSSCON device classification but with a higher computational capacity, it does not focus on the main objective of security capabilities.
- The paper “A Review of Low-End, Middle-End, and High-End Iot Devices” [5], presents a comprehensive survey of the recent and most-widely used commercial and research embedded systems and boards in different classification emphasizing their key attributes including processing and memory capabilities, security features, connectivity and communication interfaces, size, cost and appearance, operating system support, power specifications, and battery life and listing some interesting projects for each device.
- Finally, the article “Security Considerations Based on Classification of IoT Device Capabilities” [6] divides the devices in terms of its functionality and network capability, linking it with a simple threat model and security capabilities that each class is expected to have.

All these research efforts are “resource centric”, meaning that device resources (i.e., commuting power) seem and functionality are the key differentiation for dividing devices into classes. However, CROSSCON aims to have a “security centric” device classification. In that sense, CROSSCON seeks to improve on the state of the art and create a device classification that can benefit not only the CROSSCON stack specification but also other research and specifications that need a device classification with device security as central aspect.

The CROSSCON device classification is based in two aspects:

- Security Capabilities, understood as the security capabilities that the hardware and firmware of the device can offer to users and applications.
- Security Guarantees, which are security requirements a device might have due to its usage or context, and these are independent for the security capabilities. For example, a device might have very few security capabilities but require high security guarantees. This gap is therefore generating high risk exposure and presents an opportunity for improvement.

The CROSSCON device classification has been performed by investigating the security capabilities of different commercial hardware platforms such as the ARM Cortex different versions [7], x86 or RISC-V and creating a feature taxonomy with that. The security guarantees taxonomy was described to then establish device classification based on a matrix of security capability vs security guarantees. Finally, real-life examples are presented to facilitate reader comprehension and to identify potential opportunities that will drive the future CROSSCON stack design.

3.2 Security Capabilities Taxonomy

The first dimension for the Device Classification to be used in CROSSCON, is the concept of security capability. For the sake of our taxonomy, we make an important difference between the concept of security capability/feature and of security service/application. Given a specific device, we can define its security capabilities as the set of security primitives it is able to enforce over the entire or part of the system. Instead, we consider security services as more high-end implementation of a security functionalities with a specific aim, offered to the end user or to the applications of the device. For instance, memory protection and memory virtualisation are security capabilities whereas secure boot and secret key management are security services. Security capabilities can usually be one of three types:

- **Firmware based -** meaning that security is implemented entirely by software that resides in protected areas of the CPU such as the TEE. This is usually considered the less secure type of capability, due to the inclination to bugs and vulnerabilities. Actually, over the past years TEEs have been impacted by several vulnerabilities which have put many platforms and devices at risk.
- **Integrated -** meaning that security capabilities are implemented by hardware that is embedded in another unit, normally the CPU. This is more secure and more performing than firmware-based implementation, but still has a higher attack surface as the broader component that integrates the security capabilities can be exposed to more risks and vulnerabilities.
- **Dedicated or discrete hardware** isolated from any other unit on the devices. This is more secure than firmware based or integrated security capabilities, as it reduces the attack surface the unit that is implementing the security features.

It’s worth noting that the level of security is not directly mapped to these categories: although dedicated hardware is usually considered more secure, it is a general assumption that might not always be always true.

For the sake of this classification, we decide to only consider the *integrated* and *dedicated/discrete hardware* security capabilities. The aim would be to take software out of the classification: if on the one hand software extends and complements hardware-based security, on the other hand software is orthogonal to hardware and can be ported to multiple devices thus making the classification hard and not very relevant. For instance, let us consider the cryptographic capabilities of a device. These could be implemented either in software, in hardware or with a combination of the two. Therefore, it would be hard to classify a device based on whether it offers a cryptographic service in the absolute sense, since we could most likely always implement it with software. For this reason, we forsake all the security capabilities implemented in software, by either the device firmware or some dedicated library, and focus exclusively on the security-related hardware that can be leveraged to establish security. From this point forward, we will be referring to “security capability” as hardware only (integrated or dedicated). Therefore, a more capable device is not a more secure device but rather a device that can implement security services in a more efficient and most likely secure way.

For what concerns the difference between integrated and discrete hardware capabilities, we reckon that they do provide different features: while dedicated security capabilities are generally more secure given the reduced attack surface and the better isolation, on the other hand, integrated capabilities benefit from a better programmability and control. However, this is not an aspect that makes any difference in the CROSSCON stack.

We make a further distinction between the types of security capabilities based on what type of threat they protect against: hardware-based attackers and software-based attackers. Given that hardware-based attacks are an often-orthogonal problem to software-based attacks, we propose a classification focusing mostly on capabilities that hinder software-based attacks.

The security capabilities of a device are either typically provided by the CPU itself, e.g., as part of the architecture, or by additional hardware provided by the MCU manufacturers. After analysing different platforms available in the market today [8], [9], [10], [11], [12], [13], [14], [15], [16], the following capabilities have been identified:

- **Physical Tamper Detection:** Physical Tamper Detection refers to a set of techniques and mechanisms that can be used to detect and respond to physical attacks on a system. Tamper detection mechanisms can be used to detect unauthorized access, modification, or tampering with system components. Tamper detection mechanisms can be used in a wide range of applications, including embedded systems, smart cards, and secure communication systems.
- **Debug Security:** allowing developers to restrict access to debugging interfaces, preventing unauthorized access to the microcontroller's internal state and data.
- **Memory Protection:** the possibility to enforce some access control privileges on regions of the memory, for instance limiting the write/read/execute capabilities in a specific moment of the execution. This is usually enabled by a Memory Protection Unit or by a Memory Management Unit.
- **Memory Virtualization:** Normally performed by the Memory Management Unit, that provides a richer set of capabilities other than the MPU. It provides memory protection, but also functions related with virtualization of memory such as address translation, cache control, bank switching, etc. While devices that run real time operating systems such as FreeRTOS normally work in low power processors and require only MPU, devices running high end Operating systems like Linux require the usage of MMU units for full performance. We include both 1-stage and 2-stage virtualisation capabilities.
- **Secure Identifier:** unique and protected identifier that can be used to establish secure boot techniques or generate secure keys. It is usually offered by physically unclonable functions (PUFs), Unique Identifier (UID) or Composite Device Identifier (CDI). Although these offer different security levels, we group them under the same category.
- **System Level Memory Protection:** The possibility to enforce system wise memory protection that can be presented in several forms, such as:
  - write protected One-Time-Programmable or secure ROM to save secure firmware that is physically protected from unauthorised write accesses. It can be used to establish secure boot.
  - Read-protection memory, which is a memory area that cannot be accessed via software, unless specific conditions are met, or that can only be executed without read access. (e.g., product state of STM, Readout Protection STM and Secure Hide Protection, or Protected Flash Region PFR of NXP).
  - IO Virtualization is often achieved using Input/Output MMU (IOMMU) and System MMU (SMMU) hardware devices, which provide virtual memory addressing capabilities for I/O devices. This

improves system security by providing efficient restricted memory access for I/O devices, preventing unauthorized access to system resources, but also simplifies the allocation of devices to applications and virtual machines. They are commonly used in virtualization and high-performance computing environments where multiple operating systems or applications are running on a single system and need to access I/O devices simultaneously.

- IO Protection: Normally performed by a bus level access controller, IO Memory Protection ensures that memory regions accessed by a device, possibly including the CPU, follow established memory protection access control policies.

- **Cryptography Services:** dedicated cryptographic hardware modules to offer common algorithm/functionalities such as true random number generators, hash algorithms, symmetric and asymmetric key generation and verification as well as data encryption/decryption. This can include the ability to encrypt and decrypt code and data on an external/internal memory area 'on-the-fly' without requiring further operations or high latency and used for IP protection (e.g. NXP PRINCE or STM On-the-Fly description).
- **Privilege System:** the CPU provides multiple privilege execution levels that can be used to establish different security domains within the software. An example could be the privileged and unprivileged ARM execution modes.
- **Application Sandboxing**: a sophisticated isolation level for specific parts of the system that can be used to establish strong and fully programmable hardware-defined secure enclaves that isolates applications data from other processes by encrypting data in use and enforcing access controls on that application specific memory regions. This for example can be done with Intel SGX.
- **System Virtualization:** System virtualization is often achieved using a two Stage MMU. System virtualization enables a higher privilege software referred to as hypervisor, or virtual machine monitor (VMM), to allocate the physical resources of the host machine to virtual machines running on it. Each virtual machine operates independently of each other, running its own operating system and applications.
- **Application-flow protection:** techniques that restrict the control flow and/or the data flow of an application using dedicated instructions (e.g. PAC, BTI, MTE) or dedicated hardware systems (e.g. Shadow Stack or STM firewall).
- **Security Co-processor**: set of security primitives and security hardware that is separated from the main core, thus enabling deeper isolation (e.g. TPM, EdgeLock Enclave) with an error-free implementation of higher-level security services.
- **Peripheral Protection** allows peripheral access to be restricted to specific code or memory regions. This prevents unauthorized access to peripherals such as timers, GPIOs, DMAs and communication interfaces.
- **Side-Channel Protection:** Side-channel attacks are a class of attacks that exploit unintended channels of communication, such as power consumption, electromagnetic emissions, and timing information, to extract secret information from a system. Side-channel protection refers to a set of techniques and countermeasures that can be used to prevent or mitigate side-channel attacks. Side-channel protection can be implemented at different levels in the system stack, including the hardware, firmware, and user application layers.
- **Secure Monitoring:** techniques that detect faults, errors or abnormal behaviour and ultimately decide whether computations within the platform are executing as expected (e.g. NXP Secure Monitor). It can also detect tampering attempts by monitoring voltage, clock, temperature and other indicators. In the most advanced cases they can use ML algorithms (normally offloaded to integrated GPUs) for detection, based for example on hardware telemetry, potential advanced threats, such as Intel Threat Detection Technology.

Most of the capabilities presented here are common to resource constrained devices. If, on the one hand, we created a list of capabilities following the different devices present in the market, predominantly ARM based, we reckon that these capabilities are architecture agnostic. While some of them are provided directly by the reference architecture, others are vendor-specific and thus independent from the architecture itself. We believe these capabilities can be found on devices belonging to different architectures.

Finally, we assume that these capabilities can be used to implement security services for devices, such as:

- **Secure Boot**: that ensures that the code executing on the microcontroller is trusted and has not been tampered with. This is achieved using digital signatures and hash algorithms to verify the integrity of the code before it is executed.
- **Secure Storage:** of keys and data, that can be only accessed by the processes authorized to do so.
- **Measuring and Reporting**: to provide evidence to external entities to decide whether device operations are executing under the expected context.
- **Integrity Monitoring:** Integrity monitoring is a security technique used to ensure the correctness and consistency of software and data in a system. Additionally, memory monitoring techniques can monitor memory contents for any unauthorized modifications or changes. System-level monitoring techniques can also be employed to monitor system calls, network traffic, and other system events for any malicious or suspicious activities.
- **Control Flow Integrity:** CFI is originally a software-based security mechanism that protects against memory-based attacks such as buffer overflow, return-oriented programming (ROP), and jump-oriented programming (JOP). It can also be implemented in hardware, where the CPU provides support for CFI in the form of hardware-based enforcement of control flow integrity.

3.3 Security Guarantees Taxonomy

The second dimension for the Device Classification to be used in CROSSCON considers the security guarantees taxonomy. The selected model for considering security guarantees is STRIDE, which considers the following main risk categories:

- **Spoofing:** This category includes attacks aimed at impersonating the identity of a user or system in the network. They are a set of tactics and techniques that seek a compromise of identity management, authentication, and system authenticity.
- **Tampering:** These are threats related to the possibility of an IoT device being altered, and therefore its functionality changed.
- **Repudiation:** This group includes those incidents where changes can be made to systems whose authorship can be denied by the perpetrator.
- **Information Disclosure:** It includes all attacks aimed at stealing confidential information.
- **Denial of Service:** This category includes ransomware attacks that can stop the operation of devices and demand a ransom for their release.
- **Elevation of Privilege:** It contains any risk associated with a user through a system or device being able to perform actions for which they should not initially have permissions.

Specifically, STRIDE aims to ensure that an application or system complies with the CIA triad:

1. Confidentiality
2. Integrity
3. Availability

For example, if we focus on the use of **IoT devices in an industrial sector**, the impacts and recommendations, as well as the associated security guarantee for each type of STRIDE threat may be as follows:

Table 1. STRIDE threats impact and recommendations

STRIDE THREATS - [GUARANTEES]	IMPACT	RECOMMENDATIONS
Spoofing [INTEGRITY]	In industrial IoT, the cases where an attacker can impersonate a device and therefore alter the functioning of operations can have a relevant impact.	Robust cross-platform and cross-device authentication systems are essential to avoid these threats. Users and passwords are proving increasingly insecure, in favour of mutual authentication (the device identifies the platform, and the platform identifies the device) based on unique digital certificates. From the system security level, PUFs are becoming more and more relevant to identify and authenticate individual devices without relying on externally stored keys or credentials, which can be vulnerable to attack or theft.
Tampering [CONFIDENTIALITY]	Since some connected devices are linked to the operation of the business, protection against this type of attack is critical in the industry.	In this sense, devices with secure firmware and certified according to a standard such as IEC-62443-4 minimise this risk. This type of device raises the level of security and dramatically reduces the possibility of an unwanted agent modifying its hardware or software to alter its operation. to alter its operation.
Repudiation [INTEGRITY]	This case is especially relevant in scenarios where the attacker could be an insider (company employee working for a competitor).	In the industrial IoT, the elements most affected by the risk of repudiation are the platforms, given that they group data or device management, which could be massively altered without leaving a record of it. To minimise these risks, it is essential NOT to share users and access passwords, as well as the traceability and historical storage of any access and operations carried out on the systems.
Information Disclosure [CONFIDENTIALITY]	It includes all attacks aimed at stealing confidential information, either for industrial espionage or for sale or misuse. This scenario is relevant in the case of IoT, as devices are often unattended and can be physically scanned, or even stolen, by a third party.	It is therefore essential to ensure that digital information on devices at rest and in transit through the network is always encrypted. TEEs, TPMs, encryption accelerators, hardware security tokens are typical mechanisms to take into account to provide system information disclosure at rest.

	Industries with many dispersed connected assets, such as utilities, are particularly exposed to these risks.	
Denial of Service [AVAILABILITY]	This is probably the highest risk to industrial IoT, as what are commonly known as DoS attacks are those that can bring business continuity to a halt.	To minimise the risks of such attacks, network segmentation technologies and Backup & Recovery systems are imperative in any industrial IoT deployment. While they do not directly affect an attacker's ability to carry out a DoS attack, they do directly reduce the impact it can have. Network segmentation would prevent the number of devices involved in DoS from spreading to only one network segment. B&R systems would allow any downed device or system to be quickly restored to its previous state. In order to provide DoS protections measures can include the limitation of the access to specific hardware resources, or modern hardware or software-based security services for monitoring device usage.
Elevation of Privilege [INTEGRITY]	It contains any risk associated with a user through a system or device being able to perform actions for which they should not initially have permissions. For example, from a sensor, it makes sense that a database can be written to the cloud, but not that the entire database can be deleted.	Privilege escalation is usually caused by design flaws, which in many cases are made public or semi-public in the form of vulnerabilities. For this, tools that allow technological vulnerability monitoring, such as asset scanning or intrusion detection tools (IDS), are vital. The use of MMUs and MPUs that prevent malicious code from accessing specific system resources or modifying critical data structures reduces the risk of privilege escalation and other security threats.

3.4 Device Classes

One of the goals of CROSSCON is to create a classification for IoT devices based on the level of security they provide. However, it is not trivial to define a taxonomy when considering such a vast and heterogeneous domain. We identify two major directions for a classification: a theoretical approach for which we classify each device based on the security guarantees it provides (or the security services it can offer), or a more practical classification for which we only consider the security capabilities of a device. We reckon that the first approach is more generic and flexible, but it is extremely complex to create. Specifically, it would require a mapping between security guarantee/service and security capability, which would be debatable and strictly dependent on software. Let us take as example the *memory protection* capability: it is hard to determine what security guarantee it can provide and the strength of such guarantee, especially considering that it must follow a policy, an implementation and be placed in a bigger context.

We believe that each security capability is a mere building block to implement one or more security services/applications/policies that can ultimately provide some degree of security guarantee. Following this rationale, we opt for the second approach of classifying the devices based exclusively on the hardware security capabilities. As a consequence, it must be understood that these classes merely give an indication on how efficiently and securely some security guarantee can be provided. It is then up to the programmer to craft adequate firmware to leverage these capabilities.

In practice, our classification technique requires a careful analysis of the security capabilities of different devices provided by different manufacturers. Notably, this analysis cannot be limited to CPU architectures since a big portion of the security capabilities are architecture-independent and provided by the manufacturers themselves. Currently, we consider the following MCU manufacturers: NXP Semiconductors, STMicroelectronics, and Texas Instrument. Although more manufacturers could be considered, we believe they can either be added in a second moment or used to validate our classification. The proposed device classification for the CROSSCON project is the following:

- **Class 0 (NO SECURITY):** devices that have no built-in security capabilities at all. These are normally devices that respect ultra-low power and low-costs constrains, and are therefore not adequate to perform critical functions not being able to provide any security guarantee per-se. These devices need to rely entirely on software-based security, which makes them more vulnerable to attacks.
- **Class 1 (BASIC SECURITY):** devices that are resource constrained but which contains basic security capabilities such as memory protection via MPU and basic privilege system. While these devices may have a better secure stack than Class 0 devices, they may still be vulnerable to specific attacks. Providing certain security guarantees on them can be a complex task and require a lot of secure software development.
- **Class 2 (STRONG SECURITY)** devices which already contain integrated or discrete hardware functions with security capabilities such as secure storage, crypto services and measuring and reporting, as well as hardware-based enclaves. These can be MCU using CPUs such as Cortex M23 or M33.
- **Class 3 (EXTENDED SECURITY)** devices which typically can be used in high-security environments such as critical infrastructure, military applications, or secure communications. They have the highest level of security by incorporating he most advanced security capabilities such as subsystems to isolate specific parts of the device, True Random Number Generators (TRNG), physically unclonable functions (PUFs), or hardware-based intrusion detection.

While Class 0 and Class 1 devices could most likely be represented by devices that do not mount many vendors specific security capabilities, Class 2 and Class 3 are expected to be rich with those, boosting the CPU baseline security capabilities.

It's worth mentioning that these classes are not necessarily absolute and there may be some overlap between them. Additionally, security is a complex and evolving field, so the classification may change over time as new threats emerge and new security capabilities are developed, as the device classification system we have developed is based on current technology and security standards.

If the future, also working with WP6 members, proposing an IETF RFC for device classification based on its security capabilities will be explored. This RFC will help establish a more consistent and comprehensive approach to assessing and categorizing the security capabilities of various devices, and guide future research and development on the area. An RFC will facilitate an ongoing research and analysis to ensure that any device classification system remains relevant and effective over time. CROSSCON will look for feedback and input from relevant stakeholders and experts in the field in order to decide if and how address this potential work stream.

Challenges and limitations

One clear limitation of this approach is that it makes the classification of some hybrid devices harder, with MCU that might be considered cross-class. Another limitation is that in establishing boundaries is complex, as in most cases, a device equipped with security capabilities belonging to a higher class is also equipped with the capabilities of the lower classes (security is usually incremental) but on the other hand there might be exceptions some exceptions. There are cases where advanced security capabilities have been brought to more constrained devices to bring strong security capabilities without excessively increasing the complexity of the system. In these cases, we believe that a device should fall into a class based on its most advanced security capability.

Additionally, this taxonomy is not flexible with regards to new technologies that might arise in the future. These could either be added on top of the already existing capabilities or replace them. In these cases, it would make the classification somewhat obsolete. However, we believe that these are remote scenarios that could be integrated in the current classification or extend it with newer classes. Moreover, we reckon this issue is intrinsic to any security classification given the progress of this field and the inevitable obsolescence of its technologies.

Finally, this taxonomy does not follow any other rationale than the security policy of the specific manufacturers: if a security feature is only placed in a few high-end devices then it will fall under a high class. This is once again a practical approach that is suitable in a realistic scenario.

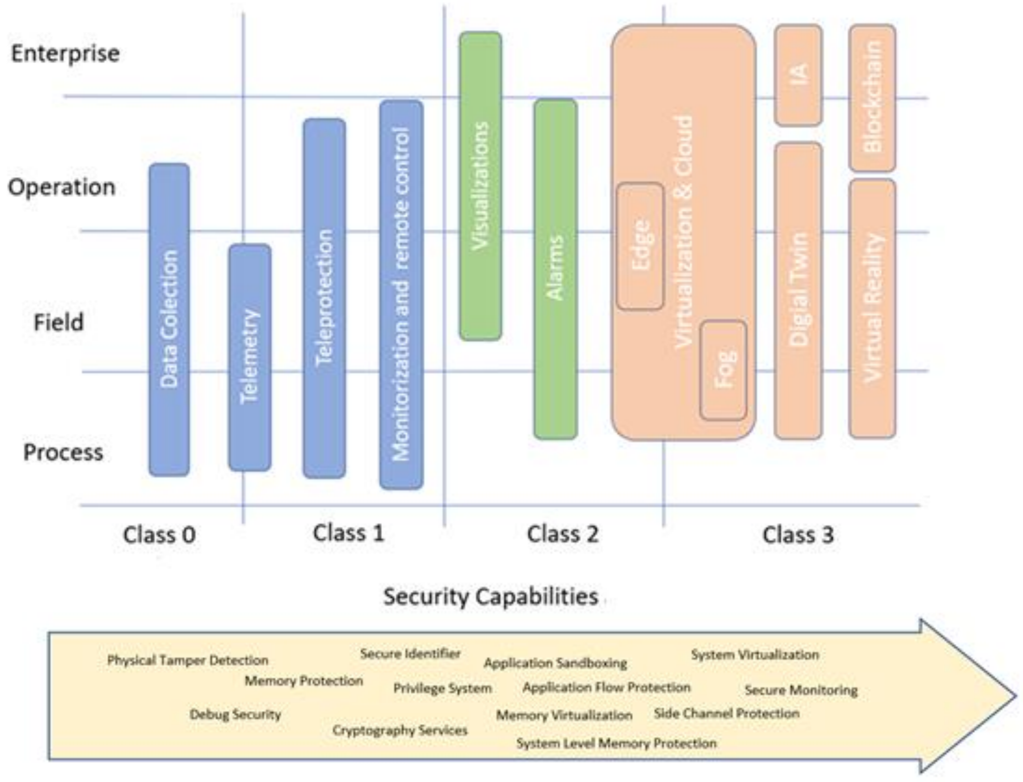


Figure 1. Device classification approach on realistic scenario

3.5 Device Examples

In this final section, real-life examples are presented to facilitate reader comprehension and to identify potential opportunities that will drive the future CROSSCON stack design.

Class 0 (NO SECURITY) devices are those that need to be designed to operate at low power, for example because they are battery-powered or have physical space limitations. Additionally, they may also have cost restrictions or are developed under tight budgets.

These restrictions imply a lack of security capabilities, so typical class 0 devices do not require specific security guarantees, and therefore used in IoT applications where security is not a major concern. Examples of class 0 devices can include Sports wearables to monitor and track fitness, simple home sensors such as smoke detectors or simple home appliances, such as smart lighting system to control the brightness and colour of the lights.

Class 1 (BASIC SECURITY) devices are also developed for combining high-performance, low-cost and low power consumption, but they have required security guarantees and therefore are implemented with more powerful microcontrollers and higher clock speeds. These allow to have basic hardware security capabilities such as MPU and others that help implementing basic security by also adding a set of software-based security services that makes use of the available hardware security capabilities.

Class 1 devices can include a wide range of embedded systems that require basic security either because its function is not critical, or because they are not exposed to a large attack surface as they are isolated or protected by other perimeter security. For example, a Smart thermostat (e.g., Google Nest or Ecobee) that use security capabilities to protect user data and authentication credentials.

Class 2 (STRONG SECURITY) devices are devices that requires strong embedded security while offering low power consumption for longer battery life. This implies the requirement of a balance of performance and security capabilities required for **secure processing, crypto acceleration and data or peripheral protection** and other features that make them well-suited for use in secure applications.

Class 2 devices can include for example cryptocurrency hardware wallets (e.g., ZelaaPay), advanced home appliances (e.g., Xiaomi Mi Home) or industrial PLCs controller that contains security mechanisms and do not rely in perimeter security or network isolation (e.g., Siemens S7-1500).

Class 3 (EXTENDED SECURITY) devices are those devices where security it's the main target and require the highest level of certification. These can typically include for example military-grade IoT devices such as tactical radios that have physical security measures like specialized tamper-resistant hardware and are designed to operate in harsh environments.

4 Relevant Standards and Industry Alliance

There are several organizations that set out requirements and recommendations regarding cybersecurity for IoT. These are translated into various standards and implementation guides.

This section aims to look at some of these standards in order to gain an insight into the fundamental requirements that the CROSSCON stack has to meet according to its criticality.

4.1 IEC 62443

This standard is focused on the field of Industry 4.0. It is a set of standards that offers an approach to industrial cybersecurity throughout the entire life cycle of a project: from risk auditing to operations. It seeks to reduce the risks that can affect assets in industrial environments.

These standards are classified into 4 blocks:

- **General:** they cover fundamental concepts, reference models and terminology.
- **Policies and procedures:** these provide guidance on the construction of a cybersecurity management programme.
- **System:** includes protection technologies and requirements to achieve a given level of security.
- **Components:** Cybersecurity technical requirements in the product development lifecycle. product development lifecycle.

The IEC 62443-4-1 certification specifies process requirements for the secure development of products.

The IEC 62443-4-2 standard addresses the security of the components (hardware and software) that have to be integrated into industrial automation and control systems. This standard differentiates between four types of components found within an industrial control system:

- **Software applications**, such as SCADA or anti-virus.
- **Embedded devices**, such as PLCs, DCS and IEDs (Intelligent Electronic Devices).
- **Host devices**, where engineering stations, data historian and operations computer stand out.
- **Network devices**, such as firewalls, switches and routers.

Following the level 1 of the IEC 62443-4-2 standard, the CROSSCON stack should be able to integrate advanced security capabilities such as device identity management, encrypted configuration and storage and secure OTA updates for all software components on a device.

Also, to be compliant with the IEC 62443-4-1 standard, the CROSSCON stack MUST follow the "cybersecurity by design" approach highlighted in the IEC standard, embedding cybersecurity best practices in its DNA and ensuring safety at every stage of the product life cycle. This product life cycle includes:

- **Integrated hardware and software:** include third-party software or applications. Isolated execution is a good practice.
- **Cybersecurity by design:** have a good design phase with threat model defined and a good requirement elicitation approach.
- **End-to-end cybersecurity:** it refers to cybersecurity features from edge to cloud, including communication protocols and integration platforms. This could be approached with the Device Management Platform proposed for the testbed of the Use Cases.

4.2 ENISA

ENISA stands for European Union Agency for Cybersecurity. As described in its web page, "it is the Union's agency dedicated to achieving a high common level of cybersecurity across Europe. Established in 2004 and strengthened by the EU Cybersecurity Act, the European Union Agency for Cybersecurity contributes to EU cyber policy, enhances the trustworthiness of ICT products, services and processes with cybersecurity certification schemes, cooperates with Member States and EU bodies, and helps Europe prepare for the cyber challenges of tomorrow. "[17]

ENISA has a series of studies, papers and guidelines concerning cybersecurity, security standards and best practices with different scenarios such as Smart Homes, Smart Manufacturing, among others.

Of the documents analysed, we believe that the ones that can help us to have a better vision of the best practices and requirements to be considered in the CROSSCON stack are those detailed below, which refer to IoT and its supply chain, IoT applied to critical infrastructures and finally, at hardware level to reinforce the CROSSCON approach.

4.2.1 ENISA Secure Supply Chain for IoT

The structure of this document provides a first overview of the IoT supply chain to identify threats and conclude with recommendations for good practice.

The first overview looks at the different stages in the IoT supply chain including:

1. Product Design
2. Semiconductor Fabrication
3. Component Manufacturing
4. Component & Embedded Software Assembly
5. Device Programming
6. IoT Platform Development
7. Distribution & Logistics
8. Service Provision & End-User Operation
9. Technical Support & Maintenance
10. Device Recovery & Repurpose

In this sense, to obtain requirements that can be focused on the CROSSCON stack, we have mainly focused on the first three steps (Product design, semiconductor fabrication and component manufacturing). Also, the Service Provision & End-User Operation section has been addressed with the Use Cases defined in D1.1[29], and an IoT platform is proposed as Barbara's platform to operate remotely on the device itself and to be able to perform the proposed UCs such as firmware update or commissioning/decommissioning.

In terms of threats to the IoT supply chain, the document refers to:

- Physical attacks
- Loss of intellectual property
- Abuse activity
- Loss of information

Finally, we have analysed the good practices for improving security in the levels of Actors, Processes and Technologies (referred in the document) that can be considered most significant for the CROSSCON stack:

Table 2. ENISA´s recommendations on improving security levels: Actors

Good practice	Threats	Supply chain stages
Develop innovative trust models	IP theft. Tampering and counterfeits.	Product Design Semiconductor Fabrication Component Assembly + Embedded Software
Promote IoT security awareness for users	Technological evolution during device life cycle.	Service Provision & End-user Operation Device Recovery
Provide security promises to customers	Majority of threats	Service Provision & End-user Operation Device Recovery

Table 3. ENISA´s recommendations on improving security levels: Processes

Good practice	Threats	Supply chain stages
Adopt security by design principles	Compromise of network. Use of factory authentication settings.	Product design. IoT platform development
Establish and improve data collection, measurement technologies, and data management	Undetected software or hardware disruptions of the devices.	Product design. Component Assembly + Embedded Software IoT platform development End-user Operation
Identify third-party software	Use of unpatched devices and systems. Implications due to standard and regulation non-compliance	Product design. Component Assembly + Embedded Software IoT platform development Device programming
Implement factory settings that use security by default	Use of factory authentication settings. Exploitation of debug interfaces. Failure of recovery procedures	Product design. Component Assembly + Embedded Software IoT platform development Device programming
Use secure data removal techniques	IP theft Use of recovered components	Device Recovery & Repurpose.

Table 4. ENISA´s recommendations on improving security levels: Technologies

Good practice	Threats	Supply chain stages
Integrate identity management systems for IoT devices	Disruptions in cloud services. Undetected software or hardware disruptions of the devices	Product design. Component Assembly + Embedded Software IoT platform development Device programming
Integrate a strong root of trust	Malware insertion. Tampering and counterfeits.	Product design. Component Assembly + Embedded Software
Implement mechanisms for remote update	Disruptions in cloud services. Undetected software or hardware disruptions of the devices	Product design. Component Assembly + Embedded Software IoT platform development Device programming
Integrate authentication mechanisms into circuits	Malware insertion. Tampering and counterfeits.	Product design. Component Assembly + Embedded Software

4.2.2 ENISA Baseline security recommendations for IoT in the context of Critical Information Infrastructures

This paper focuses on three main attack scenarios, as follows:

1. IoT administration system compromise
2. Value manipulation in IoT devices
3. Botnet/Commands injection

Focused on these scenarios, it contemplates security measures and good practices, among which we highlight the following:

- **Security by design policies:**
  - Design architecture by compartments to encapsulate elements in case of attacks.
  - For IoT hardware manufacturers and IoT software developers it is necessary to implement test plans to verify whether the product performs as it is expected. Penetration tests help to identify malformed input handling, authentication bypass attempts and overall security posture.
- **Privacy by design policies:**
  - For IoT hardware manufacturers and IoT software developers it is necessary to implement test plans to verify whether the product performs as it is expected. Penetration tests help to identify malformed input handling, authentication bypass attempts and overall security posture.
- **Asset Management Policies:**
  - Maintain procedures and configuration controls for key network and information systems (gateways, endpoint devices, networks, service platforms, etc.).
- **Hardware security measures:**
  - The Root of Trust should then be attestable by software agents running within and throughout the infrastructure.
  - Obtain hardware design with security features such as specialised security chips / coprocessors that integrate security at the transistor level, embedded in the processor, that provide:
    - Chain of trust boot-loader which authenticates the operating system before loading it.

- Chain of trust operating system which authenticates application software before loading it.
- Encryption and anonymity.
- Tamper detection.
- Trusted Execution Environment. Secure Code fetching & Execution (Integrity checks).
- Code and data signatures, built during compilation and stored and verified during execution.
- A trusted storage of device identity and authentication means, including protection of keys at rest and in use.
- Protection against unprivileged accessing security sensitive code.

- **Trust and Integrity Management:**
  - The boot process initialises the main hardware components and starts the operating system.
  - Sign code cryptographically to ensure it has not been tampered.
  - Control the installation of software on operational systems.
  - Restore Secure State.
- **Secure Software / Firmware updates:**
  - Ensure the device software/firmware, its configuration and its applications have the ability to update Over-The-Air (OTA), that the update server is secure, that the update file is transmitted via a secure connection, that it does not contain sensitive data (e.g. hardcoded credentials), and that it is signed by an authorised trust entity and encrypted using accepted encryption methods, and that the update package has its digital signature, signing certificate and signing certificate chain, verified by the device before the update process begins.
  - Offer an automatic firmware update mechanism.
- **Access control:**
  - Data integrity and confidentiality must be enforced by access controls.
  - Ensure that the device cannot be easily disassembled, and that the data storage medium is encrypted at rest and cannot be easily removed.
- **Secure and trusted communications:**
  - Ensure TLS for encryption.
  - Ensure credentials.
  - Guarantee data authenticity to enable trustable exchanges.
- **Secure interfaces and network services:**
  - Avoid provisioning the same keys.
  - Ensure only necessary ports are exposed and available.
  - Implement DDiS-resistant infrastructures.
  - Ensure web interfaces fully encrypt the user session.

The above good practices have been acquired from the study of other references studied here (among others mentioned in the document). They are as follows:

- NIST SP 800-53 - System and Services Acquisition Control Family (SA)
- OWASP Security by Design Principles
- GSM Association (GSMA) - IoT Security Guidelines

4.2.3 ENISA Hardware Threat Landscape and Good Practice Guide

This document refers mainly to threats related to hardware and firmware components. It refers to the NIST standards and proposes a series of good practices that it contemplates with the identified threats.

After analysing the document, the most significant ones are presented below, which can serve as a reference for the design of the CROSSCON stack:

Table 5. NIST good practices applied to the CROSSCON stack

Good practice	Description	Threats	Target Audience
Minimal Hardware Access	Physical access to interfaces that provide	Hardware modification (physical attacks)	Developers

Good practice	Description	Threats	Target Audience
	access to sensitive device functionality (OS boot) should be removed.	Property losses Destruction of Hardware	
Lock Logical Access	Unnecessary boot options/order should be disabled	Hardware modification (physical attacks) Property losses Destruction of Hardware	Developers, Vendors
Secure Embedded Design and Development Lifecycle	Secure coding. Implement segregation of duties, least privileges, and different trust zones.	Firmware modification Remote firmware attack Property losses Malfunction Modification or denial of service	Developers
Firmware Tamper Detection	Verification of HW related ports Verification of the deployed BIOS with known-good sources	Firmware modification Traffic Sniffing Surveillance Data Tampering	Developers, Vendors
Secure Update/Modification Management (SUM)	Handle secure modifications to the firmware. Guidance NIST SP 800-89, NIST FIPS 186-3, NIST SP 800-131A The RTU should be stored in a tamper-protected way. Evaluate computer resources limits (avoid update starvation or bricking).	Firmware modification Loss of compliance	Developers
Remote Wiping WIPE	Integrating strong authentication and authorization mechanisms.	Data loss, device recovery, and legal implications.	Developers

4.3 Industrial Internet Consortium – Industrial Internet Security Framework

The Industrial Internet Security Framework [18] is a comprehensive set of guidelines developed by the Industrial Internet Consortium (IIC) to help organizations manage the security risks associated with the Industrial Internet of Things (IIoT). It is designed to provide a standardized and consistent approach to IIoT security, with a focus on addressing the unique challenges of industrial environments.

The framework consists of three main parts:

Part I examines key system characteristics, such as safety, reliability, resilience, security, and privacy and how they should be assured together to create a trustworthy system. It also explores what makes IIoT systems different from traditional IT systems.

Part II reviews security assessment for organizations, architectures, and technologies. It outlines how to evaluate attacks as part of a risk analysis and highlights the many factors that should be considered, ranging from the endpoints and communications to management systems and the supply chains of the elements comprising the system.

Part III covers the functional and implementation viewpoint. It describes best practices for achieving confidentiality, integrity, and availability. It describes security building blocks for policy, data, endpoints, communications, monitoring, and management.

The IIC framework emphasizes the importance of implementing security controls that are tailored to the specific needs and risks of each IIoT system, and of leveraging existing security standards and best practices. It also highlights the need for ongoing monitoring, testing, and response to security incidents, as well as the importance of collaboration between different stakeholders in the IIoT ecosystem.

Overall, the IIC framework provides a robust and practical approach to managing IIoT security risks, helping organizations to ensure the safety, reliability, and security of their industrial operations.

4.4 OWASP-IoT Project

The Open Web Application Security Project (OWASP) is a global non-profit organization dedicated to improving the security of software. One of their projects is focused on securing the Internet of Things (IoT), such as smart homes, wearables, and industrial systems. The OWASP IoT project strives to identify and address the security challenges encountered by IoT devices and applications. Moreover, it offers an array of resources and tools that aid developers, researchers, and consumers in enhancing the security of IoT devices.

The OWASP IoT project guides secure design and development of IoT devices and applications, as well as testing and assessment methodologies to evaluate their security posture. The project also maintains a list of the top 10 IoT vulnerabilities and security risks, such as weak authentication, insecure communication, and lack of security updates, to raise awareness and encourage mitigation efforts. Additionally, the OWASP IoT project hosts training and educational materials, such as workshops, webinars, and documentation, to help stakeholders understand the unique security concerns of IoT and how to address them. Overall, the OWASP IoT project plays a crucial role in promoting the security and trustworthiness of the rapidly growing IoT ecosystem.

To enhance the security of IoT systems and ecosystems, the OWASP IoT project has developed various initiatives, one of which is the Internet of Things Security Verification Standard (ISVS). The ISVS is a collaborative effort that aims to establish a comprehensive and open standard of security requirements for IoT ecosystems. These requirements can be utilized throughout the development life cycle, including design, development, and testing, to ensure that IoT systems are secure and trustworthy.

IoT ecosystems can be challenging to secure due to their complexity and interconnectivity. To tackle the challenge of securing IoT ecosystems, the ISVS establishes clear and comprehensive security requirements for various components such as hardware, software, embedded applications, and communication protocols. In addition, the ISVS provides general requirements for the IoT ecosystems where these systems operate, and it relies on established industry-accepted standards when possible. By adhering to the ISVS, developers and organizations can mitigate the risks associated with IoT systems and construct more secure and durable IoT ecosystems.

The security requirements of the ISVS can be organized into a stack, starting with the foundational hardware platform (V5), which consists of various hardware components that make up the connected device. Building upon the hardware platform, the Software Platform (V3) and Communication (V4) requirements enable the development of sophisticated applications. The user space applications requirements layer (V2) provides specific requirements for these applications. Lastly, the IoT Ecosystem chapter outlines the requirements that connect the device to its broader environment, serving as the "glue" between the device and its ecosystem (V1). Overall, the ISVS stack provides a comprehensive set of security requirements for IoT systems, spanning from the device hardware to the broader ecosystem in which it operates.

The Internet of Things Security Verification Standard (ISVS) establishes a comprehensive approach to IoT security, with three levels of verification for each security requirement.

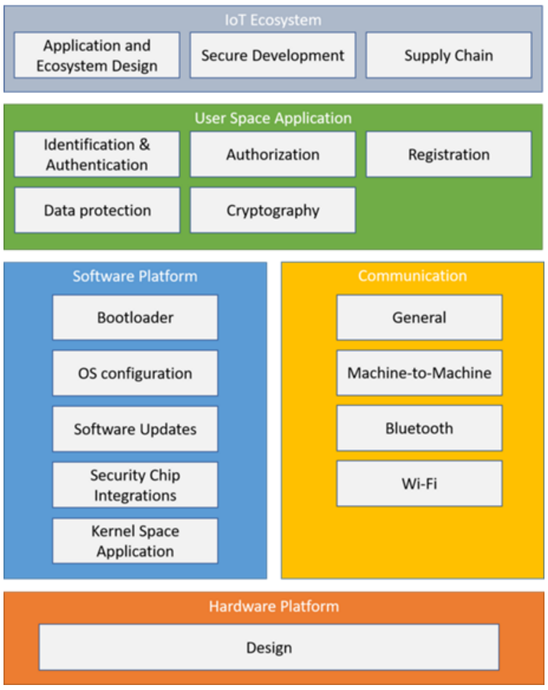


Figure 2. ISVS Stack Overview [19]

These levels are designed to build upon one another, increasing in depth as they progress. Each level includes a set of requirements that are mapped to security-sensitive capabilities and features, providing a structured approach to assessing and improving the security of IoT devices and ecosystems.

ISVS Security Verification Level 1

The primary objective of level one requirements is to safeguard against software-based attacks that do not consider physical access to the device. These requirements establish a foundational level of security for connected devices that are not considered high-risk. Such devices may not require IP protection or store sensitive information, and their compromise should not enable an attacker to move laterally to other devices or systems within the IoT ecosystem. By meeting these requirements, manufacturers can ensure that their products are less susceptible to common cybersecurity threats.

A smart light bulb is a typical example of a level one device, which typically uses off-the-shelf components and lacks advanced technology that an attacker could access through compromise. Moreover, such devices typically do not store any personal data, meaning that an attack would primarily be limited to the attacker spoofing or reading the compromised light bulb's status.

ISVS Security Verification Level 2

Level two requirements aim to protect against attacks that target the physical components of a device, going beyond software-based attacks. Devices that meet these requirements are considered critical and must be protected from compromise. Such devices typically store sensitive information and require a reasonable level of IP protection.

Level two devices include those used for security-critical tasks, such as smart locks and alarm systems, as well as devices that handle sensitive information, such as medical devices that collect patient measurement data.

ISVS Security Verification Level 3

Devices that store highly sensitive information or where a compromise could result in significant harm require the highest level of protection. These devices fall under level three requirements, which focus on preventing compromise by any means necessary. In addition to meeting the level two security requirements, level three requirements include advanced techniques to prevent reverse engineering and physical tampering.

Examples of level three devices include hardware crypto wallets, smart meters, connected vehicles, or medical implants.

4.5 GSMA Endpoint Security Guidelines

The GSM Association (GSMA) guidelines for IoT security provide recommendations to mitigate common vulnerabilities and weaknesses in an IoT ecosystem [20]. It aims to guide service providers, device manufacturers, developers, and network operators to evaluate the security of their components and services.

At first, the challenges that are being addressed in the guidelines are defined, namely availability, identity, privacy, and security. Specifically, in the context of IoT, availability focuses on lightweight network protocols and the rapid expansion of the ecosystem, identity on the identification of endpoints or services, privacy emphasizes the deployment and usage of IoT devices in an additional physical manner instead of a solely digital one, and security concentrates on specific security needs in software and hardware. Eventually, the IoT model is being defined as an ecosystem that consists of services that process data from endpoints, which are composed of low-end and rich devices, together with gateways that connect the physical devices to the digital world.

By providing checklists for service and endpoint security requirements concerning the mentioned challenges, the risks may be assessed, and one can deploy countermeasures for vulnerabilities or adapt the architecture. Additionally, the guidelines consider and give recommendations for a root of trust, the detection of abnormal behaviour, and security incidents.

Furthermore, network operators and enterprises/organizations that develop novel services that utilize cellular networks are being addressed to provide a robust and secure network.

4.6 IoT Security Foundation - IoT Security Assurance Framework

The IoT Security Assurance Framework [21] is a set of guidelines developed by the Internet of Things Security Foundation (IoTSF) to help organizations ensure the security of their IoT systems. It provides a structured approach to assessing and addressing IoT security risks, with a focus on regulatory compliance.

The framework is divided into four main sections: Governance, Baseline Requirements, Security Capability Assessment, and Security Compliance.

The Governance section outlines the policies and procedures necessary to manage IoT security risks at the organizational level. The Baseline Requirements section specifies the minimum-security controls that should be in place for all IoT systems, regardless of their specific use cases. The Security Capability

Assessment section provides a methodology for evaluating the security capabilities of IoT systems and identifying areas for improvement. The Security Compliance section provides guidance on how to demonstrate compliance with relevant regulations and standards.

The 233 requirements are divided into mandatory and advisory categories and are applicable to different device classes based on the potential impact of a compromised device. Class 0 devices are those where a hack would cause minor inconvenience, while Class 4 devices are those where a hack could cause severe consequences. The Framework covers a wide range of IoT domains, from low-value data processing to systems with high-value data and potential for significant impact.

The IoTSF framework emphasizes the need for a holistic approach to IoT security, with a focus on identifying and addressing risks at every stage of the IoT lifecycle. It also stresses the importance of collaboration between different stakeholders, including IoT manufacturers, service providers, and end-users, to ensure that security risks are effectively managed.

4.7 Regulations

Several EU specifications regarding IoT security are in place, and below are some specific examples.

4.7.1 The Radio Equipment Directive

The EU updated its Radio Equipment Directive (RED) [22] in October 2021 to include new requirements for IoT device security, which will become mandatory from August 2024. The updated directive focuses on improving network resilience, protecting consumer privacy, and reducing the risk of fraud. The requirements include preventing communication disruption, safeguarding personal data and privacy, and improving authentication for monetary transactions. Additionally, the EU published a Commission Implementing Decision in November 2022 to harmonize standards for radio equipment and support the RED. The decision is currently in effect and implements harmonized standards in support of the directive.

It also specifies that radio equipment must be designed in a manner that guarantees the protection of individuals' and domestic animals' health and safety, safeguards property, and ensures an appropriate level of electromagnetic compatibility.

4.7.2 The GDPR Directive

The General Data Protection Regulation (GDPR) [23] is a regulation passed by the European Union in 2016 that took effect on May 25, 2018. It is considered one of the most comprehensive privacy regulations in the world and applies to all businesses that collect, process, or store personal data of EU residents, regardless of the location of the business.

The goal of GDPR is to give EU citizens more control over their personal data and to unify data protection laws across the EU. It replaces the Data Protection Directive 95/46/EC and brings significant changes to the way organizations must collect, store, process, and protect personal data.

Some of the key provisions of GDPR include the requirement for organizations to obtain explicit consent from individuals before collecting or processing their personal data, the right of individuals to access their personal data, the right to be forgotten, the requirement for organizations to implement appropriate technical and organizational measures to protect personal data, and the obligation to report data breaches to data protection authorities within 72 hours.

Non-compliance with GDPR can result in significant fines, up to €20 million or 4% of global annual revenue, whichever is higher. Therefore, it is essential for businesses that collect or process personal data of EU residents to comply with GDPR and implement appropriate data protection measures.

4.7.3 The Network and Information Security Directive

The Network and Information Security (NIS) Directive [24] was adopted by the European Union in May 2016 with the aim of enhancing cybersecurity within the EU. The directive was created to ensure that EU member states are able to respond effectively to cyber threats and incidents, and to strengthen the overall cybersecurity of critical national infrastructure and essential services, including digital services providers.

The NIS Directive specifies high-level cybersecurity requirements for operators of essential services, including energy, transport, banking, financial market infrastructures, health, and water supply. It also applies to digital service providers, including online marketplaces, search engines, and cloud computing services. The directive requires member states to establish a national framework for network and information security, and to designate a competent authority responsible for overseeing the implementation of the directive within their country.

The directive includes a number of key provisions aimed at improving cybersecurity across the EU. These include the requirement for operators of essential services and digital service providers to take appropriate technical and organizational measures to manage cybersecurity risks and prevent cyber incidents. The directive also requires these organizations to report significant cyber incidents to the relevant national authorities.

In May 2022, a new legislative proposal, NIS2, was agreed upon by the EU. NIS2 builds on and will replace the existing directive. It applies to a broader scope of sectors and companies and aims to modernize the legal framework to consider the increased digitization of the internal market and the evolving cybersecurity threat landscape. NIS2 came into force on January 16, 2023, and member states will have 21 months to transpose it to their national legislative framework. The EU Agency for Cybersecurity (ENISA) will continue to support the implementation of the NIS Directive and NIS2.

4.7.4 The Cybersecurity Act

The Cybersecurity Act [25] is a regulation introduced by the European Union (EU) in 2019 to strengthen cybersecurity in the region. The Act aims to enhance the security and resilience of digital networks and information systems across the EU and improve the EU's ability to respond to cyber threats.

The Cybersecurity Act established the European Union Agency for Cybersecurity (ENISA) as the key EU agency responsible for cybersecurity matters. The Act also created a framework for European cybersecurity certification, which provides a voluntary system for products, services, and processes to demonstrate their cybersecurity credentials.

The certification framework aims to promote trust and confidence in digital products and services, as well as provide assurance to consumers and businesses that they meet appropriate cybersecurity standards. The certification process is overseen by the ENISA and includes different levels of assurance depending on the product or service being certified.

The Cybersecurity Act also established a European Cybersecurity Certification Group (ECCG) composed of representatives from EU member states to advise the European Commission on the strategic direction of the European cybersecurity certification framework.

In addition to the certification framework, the Cybersecurity Act established a cooperation group on cybersecurity, which aims to promote coordination and cooperation between EU member states on cybersecurity matters. The cooperation group also provides guidance on cybersecurity policy and initiatives.

Overall, the Cybersecurity Act represents a significant step forward in improving cybersecurity within the EU. It provides a framework for cooperation and collaboration between member states, establishes a certification scheme to promote cybersecurity best practices, and strengthens the role of the ENISA in providing guidance and support to member states.

4.7.5 Medical Device Regulation

The Medical Device Regulation (MDR) [26] is a regulation that was adopted by the European Union (EU) in 2017 and entered into force on May 26, 2021. It replaces the Medical Device Directive (MDD) and the Active Implantable Medical Devices Directive (AIMDD), which were in force since the 1990s. The MDR is designed to ensure a higher level of safety and effectiveness of medical devices that are placed on the EU market.

The MDR applies to a wide range of medical devices, including products that are used for diagnosis, treatment, or prevention of diseases, or for monitoring a patient's health. It also applies to accessories and software that are intended to be used with medical devices. The regulation covers all stages of the medical device lifecycle, from design and manufacturing to distribution and post-market surveillance.

One of the key features of the MDR is the introduction of more stringent requirements for clinical evidence. Manufacturers of medical devices are required to provide clinical data demonstrating the safety and effectiveness of their products before they can be placed on the market. The regulation also requires manufacturers to continuously monitor the safety and performance of their devices once they are on the market, and to report any adverse incidents to the relevant authorities.

The MDR also introduces new rules on the classification of medical devices, with the aim of ensuring that higher-risk devices are subject to stricter regulatory oversight. The regulation includes a new classification system based on the potential risk to patients, with higher-risk devices subject to more stringent requirements.

The MDR also strengthens the role of notified bodies, which are organizations designated by national authorities to assess the conformity of medical devices with regulatory requirements. Notified bodies are required to demonstrate their competence and independence and are subject to more stringent oversight by national authorities.

Overall, the MDR represents a significant update to the regulatory framework for medical devices in the EU. It aims to ensure a higher level of patient safety and to provide greater transparency and accountability throughout the medical device supply chain.

4.7.6 The Cyber Resilience Act

The Cyber Resilience Act [27] is a proposed regulation aimed at enhancing cybersecurity requirements for products with digital elements, to ensure more secure hardware and software products. The Act aims to address the increasing number of cyberattacks on hardware and software products that add costs for users and society. These products suffer from two primary issues: a low level of cybersecurity and a lack of information for users to select products with adequate cybersecurity properties or use them securely. Most hardware and software products are not covered by EU legislation on cybersecurity. The Act aims to create conditions for the development of secure products and allow users to consider cybersecurity when selecting and using products. The objectives of the Act include:

- ensuring manufacturers improve the security of products from design to the entire life cycle,
- facilitating compliance for hardware and software producers,
- enhancing the transparency of security properties of products, and
- enabling businesses and consumers to use products with digital elements securely.

4.7.7 European Telecommunications Standards Institute

The European Telecommunications Standards Institute (ETSI) [28] is a non-profit and independent organization that develops standards for information and communication technologies (ICT) in Europe. Its primary role is to develop harmonized European standards to support European regulation and legislation. Despite being a European organization, it has a global perspective and impact, with over 900 members from more than 60 countries, including many outside the EU.

Apart from its involvement in developing standards for areas such as edge computing, low-throughput networks, and next-generation protocols, ETSI has also been active in developing standards for consumer IoT security. In June 2020, ETSI released the first globally applicable standard for consumer IoT products, EN 303 645. This standard was developed from a standard drafted by TC CYBER, an ETSI technical committee, and from the UK government's Code of Practice for Consumer IoT Security.

The EN 303 645 was designed to establish a security baseline for connected consumer products and prevent large-scale attacks on smart devices. It specifies high-level security and data protection provisions for consumer IoT devices that are connected to network infrastructure (such as the internet or home network) and their interactions with associated services. These associated services refer to digital services, like mobile applications or third-party application programming interfaces (APIs) that are essential to provide the intended functionality of the overall consumer IoT product. However, ETSI defines these services as out of scope and focuses more on the device. The standard includes 13 recommendations:

- No universal default passwords.
- Implement a means to manage reports of vulnerabilities.
- Keep software updated.
- Securely store sensitive security parameters.
- Communicate securely.
- Minimize exposed attack surfaces.
- Ensure software integrity.
- Ensure that personal data is secure.
- Make systems resilient to outages.
- Examine system telemetry data.
- Make it easy for users to delete user data.
- Make installation and maintenance of devices easy.
- Validate input data.

It also includes a specific section on five data protection provisions for consumer IoT, intended to be supplemental to GDPR legislation and focusing on data protection from a technical perspective.

The fact that this globally used standard was only released in 2020 reflects the rapid development of the IoT market and the recent focus on addressing security issues in this fast-moving space, which is also one of the reasons that the global IoT security regulatory landscape is so fragmented.

5 Threat Analysis

The goal of this chapter is to perform the threat analysis of the Use Cases described in the D1.4[29] document, to help us define security requirements for the CROSSCON stack.

5.1 Methodology

In the following sections, we will proceed with a similar methodology for each Use Case. The first step involves identifying critical Assets of the given Use Case. An asset can be a hardware component, software, or a communication channel.

The next step is to identify the Damage Scenarios that are connected with given assets and Security Properties. By a Damage Scenario, we mean a consequence of a compromised Security for a given Asset. We can then define Threat Scenarios, which lead to compromising these Security properties. A Threat can stem from various sources, including adversarial, accidental, and environmental factors.

We conclude with the proposal of mitigation techniques for specific Threats, which can help us to define requirements in the next chapter.

5.2 Use Case 1

The first use case: Device Multi-Factor Authentication was described in detail in chapter 3 of the D1.4[29] document. This section builds on the 3.3 Threat Model to analyse the threats and define mitigation techniques and requirements.

5.2.1 Assets Identification

For MFA using PUF, the Challenge-Response Pair (CRP) table is an asset that must be protected. It is created during some kind of provisioning, during which a verifier (i.e., a device that verifies identity of another device) issues a random challenge, which is passed to prover's PUF, and whose response is sent back to the verifier. By repeating this several times, the verifier obtains a mapping of challenges and responses that can be used at a later time to assert the identity of a device.

5.2.2 Threats and Mitigation Techniques

Two main threats can be assigned with authentication: either a valid device isn't recognised, or a malicious device is mistaken for a legal one. The former case is an attack against availability, while the latter impacts confidentiality of the system.

Table 6. Threats and Mitigation Techniques: UC1

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
CRP table	C	CRP table is discovered by unauthorized actors	Attacker discovers CRP table and thus can impersonate device	Use of PUF is rate-limited by device, big number of Challenge-Response Pairs per device, provision in secure environment, encrypted communication
CRP table	A	PUF is modified without reprovisioning	Attacker destroys PUF by making it return	Use PUFs that cannot be easily modified, allow for

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
			different responses for given challenges	reprovisioning the CRP table
CRP table	C	CRP table is read from verifier	An attacker can read CRP table from verifier	Secure storage, safe API
CRP table	I	CRP table is modified on verifier	An attacker can inject own CRP entries to table on verifier	Secure storage, safe API
CRP table	A	CRP table is removed from verifier	An attacker can remove CRP table from verifier, rendering devices unaprovisioned	Secure storage, safe API, redundancy

(*) C for confidentiality, A for availability, I for Integrity

Some PUFs are modifiable, meaning that device owners or developers can create new CRP tables in case the previous one was discovered by unauthorized actors, but it also opens another point of attack. To protect against unauthorised modifications, one can choose to use non-modifiable PUF, but it has its drawbacks. If having a modifiable PUF is a must, an option to force CRP table reprovisioning can be used as a compromise to protect against such kinds of attacks.

The easiest way for an attacker to obtain the full CRP table would be to eavesdrop the communication between verifier and prover during provisioning. To protect against such eavesdropping, provisioning may be performed in a controlled environment or with encrypted communication.

Another option for an attacker is to issue its own challenges to the device, and the device would respond just as it would normally respond to the verifier. By repeating this for a large number of different challenges, the attacker can build its own CRP table that overlaps with the one used by the verifier. Replies to challenges can't be turned off because that is one of key parts of PUF-based authentication, but some logic may be added to reply only when a challenge is expected, e.g., by limiting a maximal rate of replies.

CRP tables must also be protected at rest. This means that the verifier must have secure storage option. It also requires the code running on the verifier to be safe enough to not give external actors access to data.

5.3 Use Case 2

The second use case: Firmware Updates of IoT Devices was described in detail in chapter 4 of the D1.4[29] document. This section builds on the 4.3 Threat Model to analyse the threats and define mitigation techniques and requirements.

5.3.1 Assets Identification

The following assets have been identified for UC2:

- **Device Management Server (DMS)** - a server that hosts the Firmware Update Packages, manifests, and other data are required for the firmware update process.
- **Firmware Update Package (FUP)** - a package containing firmware update to be installed on the device.
- **Authentication and Authorization Mechanisms (AM)** - mechanisms used to authenticate and authorize devices, to ensure that only authorized ones can access the download Firmware Update Packages from the DM Server.

- **Network Infrastructure (NI)** - hardware and software components used to transmit Firmware Update Packages from the DM Server to Devices.
- **Cryptographic Keys (CK)** - keys used to sign/verify and encrypt/decrypt Firmware Update Packages.
- **IoT devices** - devices that receive firmware updates.

5.3.2 Threats and Mitigation Techniques

The below table presents the Damage Scenarios and Threat Scenarios identified for each asset. We have focused on the threats that directly impact the functionality of UC2: Firmware Updates of IoT Devices.

Table 7. Threats and Mitigation Techniques: UC2

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
FUP	C	Leak confidential information about the system firmware	Attacker gathers more intelligence about the system for further attacks	FUP encryption
FUP	I	Firmware update packages downloaded from malicious location	Attacker can intercept and alter content of the FUP	FUP signing/verificati on
FUP	A	Firmware update is no longer operational	Attacker can alter the FUP to disable some (or all) services	Redundant update channel
DMS	C	Steal confidential information from DMS	Attacker can exploit vulnerability in the DMS	Regularly update DMS software
DMS	I	FUP downloaded from malicious location	Attacker can replace or modify FUP stored in the DMS	FUP signing/verificati on
DMS	A	DMS is unavailable	Attacker launches a DoS attack against the DMS	Redundant update channel; DoS mitigation techniques
AM	C	Gain unauthorized access to FUP	Attacker can exploit a weakness in AM	Implement strong AM, such as MFA
AM	A	AM services are not available	Attacker launches a DoS	Implement redundancy and

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
			against AM services	failover mechanisms; DoS mitigation techniques
NI	C	Steal confidential information from NI	Attacker can eavesdrop the traffic between DMS and IoT Device	Encrypted network traffic (e.g. HTTPS)
NI	I	Installation of malicious FUP	Attackers modifies the traffic to modify FUP	Encrypted network traffic (e.g. HTTPS)
NI	A	NI unavailable	Attacker launches a DoS against NI, so the IoT Device cannot receive FUP	Redundant communication channel; DoS mitigation techniques
CK	C	Compromise authenticity and integrity of FUP	Attacker can gain access to CK	Encrypted storage; Strong authentication mechanism to access keys
CK	A	CK are lost or unavailable, preventing FUPs from being signed	CK are lost due to administration error, or attacker action	Redundancy, backup, secure storage mechanisms
IoT Device	C	Firmware information, or other data from the device, is extracted	Attacker gathers more intelligence about the system for further attacks	Secure storage on the IoT device; firmware and other data not easily extractable from the device
IoT Device	I	Malicious firmware is running on the device	Attacker can feed malicious FUP into the device	Verify firmware prior running it

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
IoT Device	A	Device is not operational after firmware update	Incorrect or malicious FUP renders device not operational	Multiple (A/B) firmware slots

(*) C for confidentiality, A for availability, I for Integrity

Many of the threats connected with the Firmware Update Packages, can be mitigated via encryption/signing (before storing them in DM Server), and decryption/verification (before installing them on the device). The success of these mitigation techniques relies on Cryptographic Keys, which should be stored securely.

Since the DM Server is also in the scope of the UC2, good practices should be applied there as well to mitigate the threats related to the attacking DM Server directly, either to intercept the Firmware Update Packages or perform a DoS attack. To further reduce the risk of accessing FUP by an attacker, strong authentication mechanisms (such as MFA from UC1) should be applied.

In case of storing any private keys (or other secrets in the device), secure storage mechanisms should be implemented, to reduce the risk of extracting these by an attacker.

5.4 Use Case 3

The third use case: Commissioning and Decommissioning of IoT Devices was described in detail in chapter 4 of the D1.4[29] document. This section builds on the 5.3 Threat Model to analyse the threats and define mitigation techniques and requirements.

5.4.1 Assets Identification

The following assets have been identified for this UC3:

- **Device Management Server (DMS)** - a server that drives the Commissioning and Decommissioning processes.
- **Commissioning Data (CD)** - necessary information and configuration parameters (such as security certificates, credentials, application configuration, and others) acquired by the IoT devices during the Application Commissioning process.
- **Authentication and Authorization Mechanisms (AM)** - mechanisms used to authenticate and authorize devices, to ensure that only authorized ones can access the Commissioning Data intended for this particular IoT Device.
- **Network Infrastructure (NI)** - hardware and software components used to transmit Commissioning Data from the DM Server to Devices.
- **Cryptographic Keys (CK)** - keys used to sign/verify and encrypt/decrypt Commissioning Data.
- **IoT devices** - devices that undergo the Commissioning and Decommissioning processes.

5.4.2 Threats and Mitigation Techniques

The below table presents the Damage Scenarios and Threat Scenarios identified for each asset. We have focused on the threats that directly impact the functionality of UC3: Commissioning and Decommissioning of IoT Devices.

Table 8. Threats and Mitigation Techniques: UC3

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
CD	C	Leak confidential information stored in CD	Attacker gathers more intelligence about the system for further attacks	CD encryption
CD	I	CD downloaded from malicious location	Attacker can intercept and alter content of the CD	CD signing/verification
DMS	C	Steal confidential information from DMS	Attacker can exploit vulnerability in the DMS	Regularly update DMS software
DMS	I	CD downloaded from malicious location	Attacker can replace or modify CD stored in the DMS	FUP signing/verification
DMS	A	DMS is unavailable	Attacker launches a DoS attack against the DMS	Redundant update channel; DoS mitigation techniques
AM	C	Gain unauthorized access to CD	Attacker can exploit a weakness in AM	Implement strong AM, such as MFA
AM	A	AM services are not available	Attacker launches a DoS against AM services	Implement redundancy and failover mechanisms; DoS mitigation techniques
NI	C	Steal confidential information from NI	Attacker can eavesdrop the traffic between DMS and IoT Device	Encrypted network traffic (e.g. HTTPS)
NI	I	Installation of malicious CD	Attackers modifies the traffic to modify CD	Encrypted network traffic (e.g. HTTPS)

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
NI	A	NI unavailable	Attacker launches a DoS against NI, so the IoT Device cannot receive FUP	Redundant communication channel; DoS mitigation techniques
CK	C	Compromise authenticity and integrity of CD	Attacker can gain access to CK	Encrypted storage; Strong authentication mechanism to access keys
CK	A	CK are lost or unavailable, preventing FUPs from being signed	CK are lost due to administration error, or attacker action	Redundancy, backup, secure storage mechanisms
IoT Device	C	CD extracted from the device	Attacker gathers more intelligence about the system for further attacks	Secure storage on the IoT device; firmware and other data not easily extractable from the device
IoT Device	I	Malicious CN provisioned on the device	Attacker can feed malicious CN into the device	Verify CN prior applying it
IoT Device	A	Device is not operational after Commissioning	Incorrect or malicious CD renders device not operational	Device can always re-commission on failure; attestation of device (CD) state

(*) C for confidentiality, A for availability, I for Integrity

Many of the threats and mitigation techniques related to the UC2 are also applicable to this scenario. A notable distinction involves potential Commissioning Data modification, which could render the device inoperable. To address this, we can implement mechanisms that allow the device to re-commission and recover from malfunctions.

These methods can be used in conjunction with remote attestation of the Commissioning Data, issuing re-commissioning requests when incorrect configurations are detected. Furthermore, employing multiple copies of Commissioning Data can increase the overall reliability and resilience of the solution.

5.5 Use Case 4

5.5.1 Assets Identification

The following assets have been identified for the fourth use case, entitles Remote Attestation for Identification and Integrity Validation of Agricultural UAVs:

- **Attestation Result (AR)** - a short-lived (i.e., not stored for long periods of time) assertion about state of device.
- **UAV Fleet List (UFL)** - a list of vehicles authorised to operate in a fleet, along with their expected measurements. It is stored by verifier(s) for as long as a device is in use, so it must be protected accordingly.
- **Operation Logs and Configuration (OLC)** - data describing the operating environment of a fleet and its devices. It may contain field maps, crop yield, UAV operating and parking zone, flight plan and other attributes of drones.

5.5.2 Threats and Mitigation Techniques

Threats can be classified into several groups, depending on what the attacker aims to achieve. One of those groups is espionage, in which an attacker may either try to directly obtain data from used devices, inject his own vehicle into the fleet or modify existing devices to gather and forward data. The data in question may be textual or numeral inventory or attributes of devices, or even video feed from drones operating in the region. Such knowledge may be used to plan other illegal activities.

The second group of threats is *theft*. Both hardware components and even entire UAVs may be stolen, and poor security of a UAV fleet may contribute to it. For example, adversaries may replace operational UAVs with cheaper dummy devices, which will replay telemetry data in a loop, in hope of fooling any inspections.

Last class of threats is focused on destruction or denial of service. Attacks in this group may consist of modifying flight plans, for example with the aim of depleting the battery or crop protection products outside of designated fields. Another example would be altering the flight path to force the UAV to crash, be it on a targeted location or not, which could cause severe danger to not only the drone, but the human operators and bystanders as well. In addition, by changing the intensity of crop protection products, adversaries are able to damage or fully destroy crop yields.

Table 9. Threats and Mitigation Techniques: UC4

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
AR	C	Communication during attestation is intercepted	Attacker obtains data exchanged during attestation to perform replay attacks	Encrypted network traffic
AR	I	False positive result of attestation	Attacker counterfeits attestation results to make rogue device appear as a valid one	Secure attestation mechanism
AR	A	False negative result of attestation	Attacker makes valid device appear as invalid one, denying its use as a result	Secure attestation mechanism
UFL	C	List of UAVs in fleet stolen	Knowledge of drones and their configuration may be used for other attacks	Secure storage

Asset	Security Properties (*)	Damage Scenario	Threat Scenario	Mitigation Techniques
UFL	I	Injection of rogue device that may be used for espionage	Unauthorised addition of vehicle to fleet	Attestation, secure storage
UFL	A	Low performance of fleet	Attacker may remove one or more vehicles from fleet, making other UAVs work with increased performance to meet the quota expected from full fleet	Secure storage
OLC	C	Theft of hardware or products	Thief plans attack based on list of inventory and crop yield	Secure storage, authentication and authorisation
OLC	I	Modification of operation region	Attacker may modify flight plan to either include neighbouring fields (to use UAVs he doesn't own to operate on his fields) or exclude part of field (to make that field not properly taken care of)	Secure storage, authentication and authorisation
OLC	I	Hiding evidence of previous thievery	Attacker may modify inventory or products list to hide information about stolen goods	Secure storage, authentication and authorisation
OLC	A	Modification of flight plans	Flight plan may be modified by attacker to reduce effective range of drones or to waste fertiliser and crop protection products	Secure storage, authentication and authorisation

(*) C for confidentiality, A for availability, I for Integrity

All of the previously noted threats can be mitigated by secure storage and strong access control based on proper authentication and authorisation mechanisms. Encryption is also important to protect vital data, especially for data in motion. For threats that assume inclusion or modification of a vehicle, secure attestation mechanism must be used in addition to other mitigation techniques.

5.6 Use Case 5

The fifth use case pertains to Secure FPGA Provisioning. Field Programmable Gate Arrays (FPGAs) serve as configurable platforms, enabling users to program and execute their hardware designs directly. These versatile platforms can be reconfigured with various hardware designs to accommodate diverse needs.

5.6.1 Assets Identification

The following assets have been identified for the fifth use case:

- **Hardware Designs (HW IP):** Hardware designs to be configured on the FPGA are provided in the form of vendor specific configuration files. These designs may be identified as Intellectual property (IP) of their owners. Thus, the security of these hardware designs must be protected.
- **Input Data and Results (Data):** data processed and produced by the hardware design running on the FPGA.

5.6.2 Threats and Mitigation Techniques

The CROSSCON platform supports applications requiring FPGA acceleration, allowing them to delegate compute-intensive tasks to the FPGA. Each FPGA on the platform can accommodate one or more independent designs. We assume hardware IP designs are already provided in their final format, i.e., FPGA configuration files, suitable for the targeted FPGA. Within this context, we explore two options:

- **Trusted Hardware Designs:** These designs can be provided by the FPGA vendor or a reputable 3rd-party IP vendor.
- **Untrusted Hardware Designs:** In this scenario, hardware designs may be supplied by users, e.g., as in the case of cloud FPGA deployment model, where clients can bring their own IP designs, or by a 3rd-party IP vendor. Recent research has highlighted potential security risks associated with the use of malicious FPGA configuration files. Remote physical attacks become feasible, including denial-of-service attacks on FPGAs, fault injection, or side/covert channel attacks. These attacks exploit vulnerabilities such as power or thermal leakage on neighboring computing devices/resources utilizing the same power supply system. Therefore, ensuring the trustworthiness of hardware designs is paramount to safeguarding against these potential security risks.

In addition to the threats posed by malicious hardware designs, we identify another threat on HW IP. Stealing HW IP after configuration on the FPGA by reading out FPGA's configuration memory and reverse-engineering its content to acquire the original HW IP. In Table 10, we summarize threats and mitigation techniques in UC5.

Table 10: Threats and Mitigation Techniques: UC5

Asset	Security Properties	Damage Scenario	Threat Scenario	Mitigation Techniques
HW IP	Confidentiality & Integrity	Prior configuration	Attacker obtains or manipulates HW IP configuration files	HW IP encryption and authentication
HW IP	Confidentiality	After configuration	Attacker reads out FPGA's configuration memory	Access control on FPGA's configuration ports
FPGA	Availability	FPGA requires a hard reset to function again	Attacker runs a denial-of-service attack on the FPGA using malicious circuits included in the HW IP	Scan HW IP for malicious primitives
Data	Confidentiality & Integrity	During processing on FPGA	Attacker may use malicious circuits included in the neighbouring HW IP to extract secret data or manipulate it	Scan HW IP for malicious primitives

Asset	Security Properties	Damage Scenario	Threat Scenario	Mitigation Techniques
Data	Confidentiality & Integrity	In storage	Attacker obtains or manipulates data in storage	Data encryption and authentication

5.7 Summary

Our analysis reveals that the proposed Use Cases can be vulnerable to various threats, such as data leakage, un-authorized access, modification of critical data, and denial of service attacks. These threats can compromise the confidentiality, integrity, and availability of the assets involved in these Use Cases.

To protect these assets, we have identified several mitigation techniques, including:

- Encryption and secure storage of sensitive data (e.g., secure storage on IoT devices).
- Implementing strong authentication mechanisms (e.g., multi-factor authentication for access management, HW IP encryption and authentication, data encryption and authentication).
- Regularly updating software to address vulnerabilities (e.g., updating both DM server and IoT Device software).
- Employing signing and verification techniques to ensure the integrity of data (e.g., signing/verification of Firmware Update Package and Commissioning Data).
- Ensuring encrypted network traffic to protect data in transit (e.g., via HTTPS).
- Implementing redundancy, backup, and failover mechanisms to maintain availability (e.g., redundant update channels, multiple firmware slots).
- Adopting DoS mitigation techniques to protect against denial-of-service attacks.
- Implementing access control on FPGA's configuration ports.
- Implementing scan HW IP for malicious primitives.

6 Process for reviewing and validating final requirements

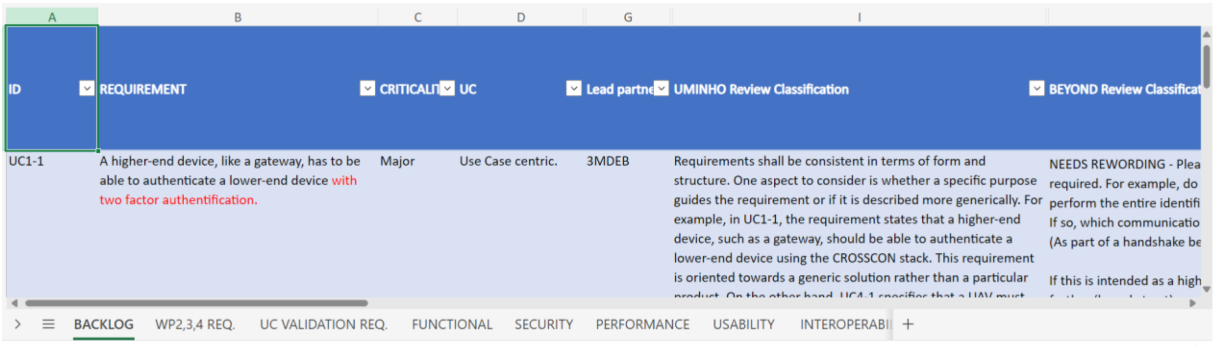
6.1 Methodology

The methodology employed for reviewing and validating requirements underscores the necessity for enhanced coordination between WP1 and WP 2, 3, and 4. To address this, a structured approach has been proposed:

1. **Coordination Meetings:**
   a. Organize bi-monthly coordination meetings where members of work packages 2, 3, and 4 present their developments to WP1.
   b. During these sessions, the focus was on refining existing requirements and identifying new ones.
2. **Approval and Comment Mechanism:**
   a. In these meetings, members of WP2, 3, and 4 will had the opportunity to approve or provide comments on requirements.
3. **Agreed Terms:**
   a. Maintain use case requirements at a higher level.
   b. Go down to a lower level of requirements around the functional requirements.
   c. Specific solutions or validation steps are to be selected later.
   d. Maintain device classification around security capabilities.
   e. Remove "should/may" to indicate criticality.
   f. Maintain a common terminology.
   g. Reference requirements. That is, there is a functional requirement and another security requirement that takes into account the same functionality.
   h. In all requirements it is important to take into account the hardware constraints for validation.
   i. UC providers are responsible for their requirements.
   j. Other validated requirements considering trusted services or specific CROSSCON stack novelty functionalities must have an implementation partner and a lead partner.
   k. WP requirements have an implementation partner. WP requirements will be correlated with UCs requirements.
4. **Working Document:**
   a. Utilize a collaborative Excel file [31] comprising:
      1. A "Backlog" tab for pending requirements awaiting approval.
      2. Other tabs for validated final requirements.

   It should be noted that after the validation process, the "backlog" tab has been deleted.

Table 11: Collaborative excel file for the requirement validation



ID	REQUIREMENT	CRITICALIT	UC	Lead partne	UMINHO Review Classification	BEYOND Review Classificat
UC1-1	A higher-end device, like a gateway, has to be able to authenticate a lower-end device with two factor authentification.	Major	Use Case centric.	3MDEB	Requirements shall be consistent in terms of form and structure. One aspect to consider is whether a specific purpose guides the requirement or if it is described more generically. For example, in UC1-1, the requirement states that a higher-end device, such as a gateway, should be able to authenticate a lower-end device using the CROSSCON stack. This requirement is oriented towards a generic solution rather than a particular	NEEDS REWORDING - Plea required. For example, do perform the entire identifi If so, which communicatio (As part of a handshake be If this is intended as a high

6.2 Main Objectives

The main objectives were:

1. **Consensus Building:**
   Achieve consensus on rewording, acceptance, or rejection of requirements through collective discussions.
2. **Validation of UC Requirements:**

   Validate user-centric (UC) requirements to provide valuable feedback for the refinement of the "Validation Criteria" deliverable.

This structured methodology aimed to foster effective communication and collaboration, ensuring a streamlined process for refining, approving, and validating requirements across different work packages.

7 CROSSCON Requirements Elicitation Tables

The following tables contain the final set of requirements that have been derived from D1.4[29] and D1.2 [30]. By incorporating feedback from a variety of use cases, we can ensure the CROSSCON results will be flexible, adaptable, and capable of meeting the evolving needs of more users over time.

However, it is acknowledged that the requirements are high level and most of them might not be directly affecting the CROSSCON stack design and development, but rather the services that are built on top of the stack. So, when a requirement says "The CROSSCON stack has to..." that could apply either directly to the low level CROSSCON stack (e.g. hypervisor, isolation and abstraction layer) or to the security services built on top of the stack (e.g. secure boot, secure storage). Also, there are some requirements designated as "Use Case" requirements that are specific requirements related to the functionality of each individual use case.

Combining all these requirements will ensure that the work to be done meets the needs of end-users brought by the UC providers and addresses the challenges they identified. However, it is important to note that these requirements have different criticalities and therefore, "CRITICAL" must be fulfilled to ensure the viability of the CROSSCON stack, "MAJOR" should be completed to achieve satisfactory outcomes, and "MINOR" can be considered optional to complete.

In this Final Technical Specification, the requirements set on the first deliverable have evolved into validated requirements with a general consensus of the implementation partners.

To match these requirements to the CROSSCON stack design, specific WP requirements have been added. These WP requirements have been mapped to the use cases to ensure the work meet user needs and project challenges. The use case partners are responsible for the implementation and validation of their requirements, while the WP partners are responsible for theirs (with some exceptions that will be specified latter on D1.6). The other functional, security, interoperability, usability and performance requirements have different implementation partners and will be further discussed during the Validation Criteria D1.6.

7.1 Use Case Validation Requirements: Final version

Table 12. Validation Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
UC1-1	A high-end device, like a gateway, has to be able to authenticate a low-end device with two factor authentication.	Major	3MDEB	UC1
UC1-2	Two high-end devices, like gateways, have to be able to mutually authenticate themselves using two factor authentication.	Major	3MDEB	UC1
UC2-1	A device has to be able to get a unique identifier (ID) that can be used to identify itself to the server.	Major	BIOT	UC1, UC2, UC3, UC4
UC2-2	The device needs to be able to download the firmware image.	Major	BIOT	UC2
UC2-3	The device needs to be able to store that information in such a way that it can only be accessed by the authorized services (that need that information) (This can be done through "secure" storage).	Major	BIOT	UC2, UC3

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
UC2-4	The update should only be applied after ensuring the update's integrity and authenticity.	Major	BIOT	UC2
UC3-1	The device needs to be able to download the provisioning information.	Major	BIOT	UC3
UC4-1	The device has a private identifier (ID) that can be used to identify itself to third parties.	Major	CYSEC	UC4
UC4-2	The device has to be able to attest the status of its system to a remote verifier. The exact attestation procedure will be determined later in the project but shall implement a remote attestation report.	Major	CYSEC	UC4
UC4-3	The user can select which measurements are included within the remote attestation report of the device from a predefined list of possible measurements.	Major	CYSEC	UC4
UC4-4	The device connects to the remote attestation server using a secure and authenticated communication channel.	Major	CYSEC	UC4
UC4-5	The device provides an attestation conclusion (accepted or rejected), depending on the response of the remote attestation server to the delivered attestation measurements. Whether or not the attestation conclusion can be overwritten by the user, and if so under which conditions, will be determined later on in the project.	Major	CYSEC	UC4
UC4-6	The device can perform a remote attestation while in motion, including when no connection to the remote attestation server can be established.	Major	CYSEC	UC4
UC5-1	The CROSSCON stack needs to mediate the access to FPGA device resources (e.g., JTAG, configuration engine, configurable logic, buses).	Critical	TUD	UC5
UC5-2	The CROSSCON stack should enable the secure configuration of functional bitstreams, i.e., hardware designs.	Major	TUD	UC5
UC5-3	The CROSSCON stack support access control to hardware design(s) on the FPGA.	Major	TUD	UC5
UC5-4	The CROSSCON stack should enable secure communication of workloads and results to different hardware designs on the FPGA.	Major	TUD	UC5

7.2 Functional Requirements: Final version

Table 13. Functional Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
FUNC-1	The CROSSCON stack has to be able to provide the device with a unique identifier (ID).	Critical	3MDEB	UC1, UC2, UC3, UC4
FUNC-2	The CROSSCON stack has to be able to provide MFA service.	Critical	3MDEB	UC1
FUNC-3	The CROSSCON stack implements a secure boot of both the device and the stack itself.	Minor	3MDEB	UC1
FUNC-4	The CROSSCON stack provides a Remote Attestation (RA) service.	Critical	CYSEC	UC4
FUNC-5	The CROSSCON stack should provide a way to configure which measurement should be included in the attestation report.	Major	CYSEC	UC4
FUNC-6	The CROSSCON stack should provide a secure storage capability.	Critical	3MDEB	UC1, UC2, UC3, UC4
FUNC-7	Some CROSSCON stack attestation measurements have to be triggered based on conditional assumptions/triggers setup by the device manufacturer. Note: Configuration parameters will be later defined.	Minor	CYSEC	UC4
FUNC-8	The CROSSCON stack has to be able to receive firmware images and to write those images to persistent storage so that they can be used by the device across reboots.	Minor	BIOT	UC2
FUNC-9	The CROSSCON stack needs to provide a mechanism that allows communication channel with authentication.	Critical	BIOT	UC2
FUNC-10	The CROSSCON stack needs to provide a mechanism that allows encrypted communication channel.	Critical	BIOT	UC2
FUNC-11	The CROSSCON stack needs to provide a mechanism that allows communication channel with message integrity.	Critical	BIOT	UC2
FUNC-12	The CROSSCON stack has to enable the secure decrypt the received firmware image.	Critical	BIOT	UC2

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
FUNC-13	The CROSSCON stack should be able to report its version.	Minor	BIOT	UC1, UC2, UC3, UC4, UC5
FUNC-14	The CROSSCON stack has to be able to guarantee the integrity and confidentiality of the information provisioned by the manufacturer.	Critical	BIOT	UC3
FUNC-15	If PUF is available, the CROSSCON stack has to offer rate-limitation of PUF usage, to avoid CRP (Challenge-Response Pair) table discovery.	Major	3MDEB	UC1
FUNC-16	The CROSSCON stack has to provide capability to act as PUF-based authentication prover and verifier, when the hardware allows it.	Critical	3MDEB	UC1
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor	BIOT	UC1, UC2, UC3, UC4, UC5

7.3 Security Requirements: Final version

Table 14. Security Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
SEC-1	The CROSSCON stack has to ensure the freshness of the attestation report.	Critical	CYSEC	UC4
SEC-2	The CROSSCON stack has to provide a mechanism to ensure integrity and authenticity of firmware image.	Major	BIOT	UC2
SEC-3	The CROSSCON stack should support isolation of environment.	Minor	BIOT	UC1, UC2, UC3, UC4, UC5
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor	BIOT	UC1, UC2, UC3, UC4, UC5
SEC-5	The CROSSCON stack should provide access to PUFs, if available.	Minor	3MDEB	UC1

7.4 Performance Requirements: Final version

Table 15. Performance Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major	BIOT	UC1, UC2, UC3, UC4, UC5

7.5 Usability Requirements: Final version

Table 16. Usability Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
UX-1	The CROSSCON stack should provide an API that allows the user to add measurements to be included in the attestation report, besides those already present in a default list of available measurements.	Minor	CYSEC	UC4
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical	BIOT	UC1, UC2, UC3, UC4, UC5
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical	BIOT	UC1, UC2, UC3, UC4, UC5

7.6 Interoperability Requirements: Final version

Table 17. Interoperability Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	USE CASE APPLICABLE
IOP-1	The CROSSCON stack should be demonstrated in two architectures and in each class of devices.	Major	3MDEB	UC1
IOP-2	The CROSSCON stack MFA service should use properties of the WiFi signal, if such wireless technology is supported by the hardware.	Major	3MDEB	UC1

7.7 Work Package Requirements

Table 18. WP Requirements

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	CROSSCON STACK
WP3-1	The CROSSCON stack SHOULD support multiple isolated execution environments.	Critical	UMINHO	CROSSCON Hypervisor

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	CROSSCON STACK
WP3-2	The CROSSCON stack SHOULD support running unmodified TAs in multiple architectures.	Critical	UMINHO	CROSSCON Baremetal TEE
WP3-3	The CROSSCON stack SHOULD guaranteed that a trusted kernel / TA cannot access arbitrary platform resources.	Critical	UMINHO	CROSSCON Hypervisor and CROSSCON Baremetal TEE
WP3-4	The CROSSCON stack SHOULD provide a global platform compliant runtime environment.	Critical	UMINHO	CROSSCON Baremetal TEE
WP3-5	The CROSSCON stack SHOULD support eventual extensions to GP internal core API.	Critical	UMINHO	CROSSCON Baremetal TEE
WP3-6	The CROSSCON stack SHOULD decomposition of trusted services from the platform TEE.	Critical	UMINHO	CROSSCON Hypervisor
WP3-7	The CROSSCON stack SHOULD support multiple TEE programming models simultaneously.	Critical	UMINHO	CROSSCON Hypervisor
WP3-8	The CROSSCON stack SHOULD allow parametrization of TEE properties.	Critical	UMINHO	CROSSCON Hypervisor
WP3-9	The CROSSCON Hypervisor SHOULD provide microarchitectural side channel mitigation.	Critical	UMINHO	CROSSCON Hypervisor
WP3-10	The CROSSCON Hypervisor SHOULD allow dynamic VM instantiation.	Critical	UMINHO	CROSSCON Hypervisor
WP3-11	The CROSSCON Hypervisor SHOULD support multiple architectures for high- and low-end class of devices (APU, MCU, RTU).	Critical	UMINHO	CROSSCON Hypervisor
WP3-12	The CROSSCON Hypervisor SHOULD allow per-VM TEE services.	Critical	UMINHO	CROSSCON Hypervisor
WP3-13	The CROSSCON Hypervisor SHOULD support multiple VMM execution.	Major	UMINHO	CROSSCON Hypervisor
WP3-14	The CROSSCON Hypervisor SHOULD mediate access to	Critical	UMINHO	CROSSCON Hypervisor

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	CROSSCON STACK
	FPGA resources, JTAG, Flashing, Buses.			
WP3-15	The CROSSCON Hypervisor SHOULD support the target platforms security mechanisms.	Major	UMINHO	CROSSCON Hypervisor
WP4-1	CROSSCON SoC SHOULD provide the necessary HW features so it can be used with the CROSSCON hypervisor.	Critical	BEYOND	CROSSCON SoC
WP4-2	CROSSCON SoC SHOULD provide the necessary HW features that allow a form of isolation between different domains (e.g. RISC-V's SPMP extension).	Critical	BEYOND	CROSSCON SoC
WP4-3	CROSSCON SoC SHOULD provide a way to integrate HW accelerators into the SoC in a way that provides isolation between different domains.	Critical	BEYOND	CROSSCON SoC
WP4-4	CROSSCON SoC MAY provide HW features that allow to reduce the size of the code (e.g. RISC-V's Zc extension).	Minor	BEYOND	CROSSCON SoC
WP4-5	CROSSCON SoC MAY provide additional protection against side channel attacks.	Minor	BEYOND	CROSSCON SoC
WP4-6	CROSSCON SoC MAY provide support for hardware accelerated control-flow attestation (e.g. trace port).	Minor	BEYOND, UNITN, TUD	CROSSCON SoC
WP4-7	Perimeter guard SHOULD allow multiple isolated domains to access the HW accelerator.	Critical	BEYOND	CROSSCON SoC
WP4-8	Perimeter guard SHOULD provide isolation between domains.	Critical	BEYOND	CROSSCON SoC
WP4-9	Perimeter guard MAY also be used to integrate other domain specific HW (e.g. peripheral devices) and not just HW accelerators.	Minor	BEYOND	CROSSCON SoC
WP5-1	The CROSSCON stack needs to enable access to FPGA device resources (e.g., JTAG,	Critical	TUD	FPGA

ID	REQUIREMENT	CRITICALITY	LEAD PARTNER	CROSSCON STACK
	configuration engine, configurable logic, buses).			
WP5-2	The CROSSCON stack should enable the secure configuration of functional bitstreams, i.e., hardware designs.	Critical	TUD	FPGA
WP5-3	The CROSSCON stack should enable access control to hardware design(s) on the FPGA.	Critical	TUD	FPGA
WP5-4	The CROSSCON stack should enable secure communication of workloads and results to hardware design(s) on the FPGA.	Critical	TUD	FPGA

8 Mapping of Requirements to Use Cases

This section aims to provide a vision of the requirements focused on each of the use cases. This will help during the testing and validation process, to have a clear idea of the needs and requirements of each defined use case.

The following table shows the mapping of the requirements and those that are specific to each use case or shared by several or all of them.

Table 19. Requirements mapped to Use Cases

ID	UC1	UC2	UC3	UC4	UC5
UC1-1	X				
UC1-2	X				
UC2-1	X	X	X	X	
UC2-2		X			
UC2-3		X	X		
UC2-4		X			
UC3-1			X		
UC4-1				X	
UC4-2				X	
UC4-3				X	
UC4-4				X	
UC4-5				X	
UC5-1					X
UC5-2					X
UC5-3					X
UC5-4					X
FUNC-1	X	X	X	X	
FUNC-2	X				
FUNC-3	X				
FUNC-4				X	
FUNC-5				X	
FUNC-6	X	X	X	X	
FUNC-7				X	
FUNC-8		X			
FUNC-9		X			
FUNC-10		X			
FUNC-11		X			
FUNC-12		X			
FUNC-13	X	X	X	X	X

ID	UC1	UC2	UC3	UC4	UC5
FUNC-14			X		
FUNC-15	X				
FUNC-16	X				
FUNC-17	X	X	X	X	X
SEC-1				X	
SEC-2		X			
SEC-3	X	X	X	X	X
SEC-4	X	X	X	X	X
SEC-5	X				
PERF-1	X	X	X	X	X
UX-1				X	
UX-2	X	X	X	X	X
UX3	X	X	X	X	X
IOP-1	X				
IOP-2	X				

Taking into account the trusted services and security primitives that will be developed during this project, the following chart includes a mapping of tursted services and security primitives to Use Cases.

Table 20. Trusted Sevices mapped to Use Cases

Trusted Services/Security Primitives	UC1	UC2	UC3	UC4	UC5
Device Authentication	X				
Secure boot	X	X	X	X	
Secure storage	X	X	X	X	X
Secure Provisioning		X	X		
Isolated Execution	X	X	X	X	X
Secure Communication	X	X	X	X	X
Remote Attestation				X	
Code and Data Integrity				X	
Crypto primitives					X
Trust anchor on the FPA					X
Up-to-date FPGA virus scanner					X
Remote secret key generation					X
Root of Trust				X	
PUF and context-based authentication	X				
Control flow integrity	X	X	X	X	X
Secure firmware update		X	X		

For a definition of **Trusted Services and Security Primitives** please refer to D2.3 Open Specification (month 26) together with the rest of the stack definitions.

8.1 Use Case 1- Final

Table 21. Use Case Requirements: UC1

ID	REQUIREMENT	CRITACALITYY
UC1-1	A high-end device, like a gateway, has to be able to authenticate a lower-end device with two factor authentication.	Major
UC1-2	Two high-end devices, like gateways, have to be able to mutually authenticate themselves using two factor authentication.	Major
UC2-1	A device has to be able to get a unique identifier (ID) that can be used to identify itself to the server.	Major

Table 22. Other Requirements: UC1

ID	REQUIREMENT	CRITACALITYY
FUNC-1	The CROSSCON stack has to be able to provide the device with a unique identifier (ID).	Critical
FUNC-2	The CROSSCON stack has to be able to provide MFA service.	Critical
FUNC-3	The CROSSCON stack implements a secure boot of both the device and the stack itself.	Minor
FUNC-6	The CROSSCON stack should provide a secure storage capability.	Critical
FUNC-13	The CROSSCON stack should be able to report its version.	Minor
FUNC-15	If PUF is available, the CROSSCON stack has to offer rate-limitation of PUF usage, to avoid CRP (Challenge-Response Pair) table discovery.	Critical
FUNC-16	The CROSSCON stack has to provide capability to act as PUF-based authentication prover and verifier, when the hardware allows it.	Critical
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor
SEC-3	The CROSSCON stack should support isolation of environment.	Minor
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor
SEC-5	The CROSSCON stack should provide access to PUFs, if available.	Minor
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical

ID	REQUIREMENT	CRITACALITYY
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical
IOP-1	The CROSSCON stack should be demonstrated in two architectures and in each class of devices.	Major
IOP-2	The CROSSCON stack MFA service should use properties of the WiFi signal, if such wireless technology is supported by the hardware.	Major

8.2 Use Case 2 - Final

Table 23. Use Case Requirements: UC2

ID	REQUIREMENT	RITACALITYY
UC2-1	A device has to be able to get a unique identifier (ID) that can be used to identify itself to the server.	Major
UC2-2	The device needs to be able to download the firmware image.	Major
UC2-3	The device needs to be able to store that information in such a way that it can only be accessed by the authorized services (that need that information). (This can be done through "secure" storage.)	Major
UC2-4	The update should only be applied after ensuring the update's integrity and authenticity.	Major

Table 24. Other requirements: UC2

ID	REQUIREMENT	CRITACALITY
FUNC-1	The CROSSCON stack has to be able to provide the device with a unique identifier (ID).	Critical
FUNC-6	The CROSSCON stack should provide a secure storage capability.	Critical
FUNC-8	The CROSSCON stack has to be able to receive firmware images and to write those images to persistent storage so that they can be used by the device across reboots.	Minor
FUNC-9	The CORSSCON stack needs to provide a mechanism that allows communication channel with authentication.	Critical
FUNC-10	The CORSSCON stack needs to provide a mechanism that allows encrypted communication channel.	Critical
FUNC-11	The CORSSCON stack needs to provide a mechanism that allows communication channel with message integrity.	Critical
FUNC-12	The CROSSCON stack has to enable the secure decrypt the received firmware image.	Critical
FUNC-13	The CROSSCON stack should be able to report its version.	Minor

ID	REQUIREMENT	CRITACALITY
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor
SEC-2	The CROSSCON stack has to provide a mechanism to ensure integrity and authenticity of firmware image.	Major
SEC-3	The CROSSCON stack should support isolation of environment.	Minor
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical

8.3 Use Case 3 - Final

Table 25. Use Case Requirements: UC3

ID	REQUIREMENT	CRITACALITYY
UC2-1	A device has to be able to get a unique identifier (ID) that can be used to identify itself to the server.	Major
UC2-3	The device needs to be able to store that information in such a way that it can only be accessed by the authorized services (that need that information) (This can be done through "secure" storage).	Major
UC3-1	The device needs to be able to download the provisioning information.	Major

Table 26. Other Requirements: UC3

ID	REQUIREMENT	CRITACALITYY
FUNC-1	The CROSSCON stack has to be able to provide the device with a unique identifier (ID).	Critical
FUNC-6	The CROSSCON stack should provide a secure storage capability.	Critical
FUNC-13	The CROSSCON stack should be able to report its version.	Minor
FUNC-14	The CROSSCON stack has to be able to guarantee the integrity and confidentiality of the information provisioned by the manufacturer.	Critical

ID	REQUIREMENT	CRITACALITYY
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor
SEC-3	The CROSSCON stack should support isolation of environment.	Minor
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical

8.4 Use Case 4 - Final

Table 27. Use Case Requirements: UC4

ID	REQUIREMENT	CRITACALITYY
UC2-1	A device has to be able to get a unique identifier (ID) that can be used to identify itself to the server.	Major
UC4-1	The device has a private identifier (ID) that can be used to identify itself to third parties.	Major
UC4-2	The device has to be able to attest the status of its system to a remote verifier. The exact attestation procedure will be determined later in the project but shall implement a remote attestation report.	Major
UC4-3	The user can select which measurements are included within the remote attestation report of the device from a predefined list of possible measurements.	Major
UC4-4	The device connects to the remote attestation server using a secure and authenticated communication channel.	Major
UC4-5	The device provides an attestation conclusion (accepted or rejected), depending on the response of the remote attestation server to the delivered attestation measurements. Whether or not the attestation conclusion can be overwritten by the user, and if so under which conditions, will be determined later on in the project.	Major
UC4-5	The device can perform a remote attestation while in motion, including when no connection to the remote attestation server can be established.	Major

Table 28. Other Requirements: UC4

ID	REQUIREMENT	CRITACALITYY
FUNC-1	The CROSSCON stack has to be able to provide the device with a unique identifier (ID).	Critical
FUNC-4	The CROSSCON stack provides a Remote Attestation (RA) service.	Critical
FUNC-5	The CROSSCON stack should provide a way to configure which measurement should be included in the attestation report.	Major
FUNC-6	The CROSSCON stack should provide a secure storage capability.	Critical
FUNC-7	Some CROSSCON stack attestation measurements have to be triggered based on conditional assumptions/triggers setup by the device manufacturer. Note: Configuration parameters will be later defined.	Minor
FUNC-13	The CROSSCON stack should be able to report its version.	Minor
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor
SEC-1	The CROSSCON stack has to ensure the freshness of the communication channel (integrity and confidentiality).	Critical
SEC-3	The CROSSCON stack should support isolation of environment.	Minor
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major
UX-1	The CROSSCON stack should provide an API that allows the user to add measurements to be included in the attestation report, besides those already present in a default list of available measurements.	Minor
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical

8.5 Use Case 5

Table 29. Use Case Requirements: UC5

ID	REQUIREMENT	CRITACALITYY
UC5-1	The CROSSCON stack needs to mediate the access to FPGA device resources (e.g., JTAG, configuration engine, configurable logic, buses).	Critical
UC5-2	The CROSSCON stack should enable the secure configuration of functional bitstreams, i.e., hardware designs.	Major
UC5-3	The CROSSCON stack support access control to hardware design(s) on the FPGA.	Major
UC5-4	The CROSSCON stack should enable secure communication of workloads and results to different hardware designs on the FPGA.	Major

Table 30. CROSSCON stack Requirements: UC5

ID	REQUIREMENT	CRITACALITYY
FUNC-13	The CROSSCON stack should be able to report its version.	Minor
FUNC-17	The CROSSCON stack should provide a mechanism that allows to run an application in the isolated execution environment.	Minor
SEC-3	The CROSSCON stack should support isolation of environment.	Minor
SEC-4	The CROSSCON stack should provide a high entropy source, if allowed by Hardware constrains.	Minor
PERF-1	The CROSSCON stack performance impact shall be tested and documented.	Major
UX-2	The CROSSCON stack has to have the ability to be updated remotely.	Critical
UX-3	The CROSSCON stack has to have a comprehensive and well documented set of APIs.	Critical

9 Conclusions

We have presented the final results of the CROSSCON requirements elicitation activity performed in T1.5.

We have first highlighted the need of a classification scheme of IoT devices, suitable to the project, that abstracts away performance-centric features and offers a suitable categorisation of the security capabilities (features) the device has, as well as the security guarantees (properties) the device offers. After due consideration and gap analysis based on the state of the art, we have come up with a new IoT device classification scheme (Sec. 2) that addresses and scales to the needs of CROSSCON. We believe this classification scheme will serve not only during the project duration but will also hold value for other initiatives. As such, we will also strive for external project valorisation and uptake.

To better position the work on requirements elicitation, we have first examined the relevant regulations, standards, and directives (Sec. 3) which set various requirements and recommendations on the cybersecurity aspects of IoT devices in different application domains and sectors. This helped us get better understanding and context on the core and essential requirements of the CROSSCON stack.

Next, we have presented the threat analysis performed on the CROSSCON use cases (Sec. 4). The study of the relevant threats, their impact on the use cases, and potential mitigation techniques greatly helped us in the understanding and specification of the initial security requirements of the CROSSCON stack. We recall that the initial CROSSCON use cases were presented in D1.1, and two more use cases in D1.4 that have been developed by the consortium (i.e., UC4 and UC5).

The technical specification of the CROSSCON requirements is given in Section 6. To achieve a continuous feedback loop from the technical activities, the results the D1.2 has served as a starting point of a live, working document on the CROSSCON requirements elicitation, that has been shared and revised on cycles (based on agile principles) according to interim results and feedback from WP2, WP3, and WP5 activities, facilitated by short focused sessions between the use case providers, and the academic and industrial partners of the consortium. This final version has considered the previous requirements but have been discussed and validated on a consensus including both "Security Stack" and "Security Services" orientation.

The requirements have been separated into 5 different groups defining functional, security, integration, performance, and usability. This set of requirements have been mapped to a trusted service, a security primitive, or to a specific CROSSCON stack component.

This final version of the requirements specification has taken important inputs from WP2, 3 and 4 and these requirements have already been added. These WP requirements are related to CROSSCON stack development and on the domain-specific hardware extension primitives, respectively.

As the project has been progressing on the specification of the CROSSCON stack in WP2, and the technical development of its components, services, toolchain, and primitives in WP2 and WP3, the requirements elicitation process has also progressed in parallel, as a continuous feedback loop from such activities.

The first version of this document raised important issues, discussions about viability and implementation that have been progressing throughout these months. This final version is now the reference point for the final validation criteria of CROSSCON and for the pilots' implementation and validation activities in WP5.

References

[1] Carsten Bormann, Mehmet Ersue , Ari Keränen, “Terminology for Constrained-Node Networks-RFC 7228”, in IESG for Publication.

[2] Uribe, Felix, “The Classification of Internet of Things (IoT) Devices Based on Their Impact on Living Things” in SSRN Electronic Journal, 10.2139/ssrn.3350094, 2017/06/01.

[3] H. Mohd Aman, E. Yadegaridehkordi, Z. S. Attarbashi, R. Hassan and Y. -J. Park, "A Survey on Trend and Classification of Internet of Things Reviews," in IEEE Access, vol. 8, pp. 111763-111782, 2020, doi: 10.1109/ACCESS.2020.3002932.

[4] A. Desai, D. M. Divakaran, I. Nevat, G. W. Peter and M. Gurusamy, "A feature-ranking framework for IoT device classification," 2019 11th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, India, 2019, pp. 64-71, doi: 10.1109/COMSNETS.2019.8711210.

[5] M. O. Ojo, S. Giordano, G. Procissi and I. N. Seitanidis, "A Review of Low-End, Middle-End, and High-End Iot Devices," in IEEE Access, vol. 6, pp. 70528-70554, 2018, doi: 10.1109/ACCESS.2018.2879615.

[6] Seungyong Yoon, Jeongnyeo Kim, Yongsung Jeon (2017), “Security Considerations Based on Classification of IoT Device Capabilities”, The Ninth International Conferences on Advanced Service Computing

[7] Demystifying Arm TrustZone: A Comprehensive Survey SANDRO PINTO, Centro Algoritmi, Universidade do Minho NUNO SANTOS, INESC-ID, Instituto Superior Técnico, Universidade de Lisboa: https://www.dpss.inesc-id.pt/~nsantos/papers/pinto_acsur19.pdf

[8] Cortex-M0+ Processor Technical Reference Manual: https://developer.arm.com/documentation/ddi0434/latest/

[9] Cortex-M3 Devices: Specifications: https://developer.arm.com/Processors/Cortex-M3

[10] Cortex-M3 Devices: Fault handling: https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-processor/fault-handling?lang=en

[11] Introduction to STM32 microcontrollers security: https://www.compel.ru/wordpress/wp-content/uploads/2019/09/en.dm00493651-1.pdf

[12] Cortex-M33 and TrustZone for Armv8-M Technical Webinar: https://www.arm.com/resources/webinars/cortex-m33-and-trustzone-for-armv8-m-technical-webinar

[13] Arm-based processors: https://www.ti.com/microcontrollers-mcus-processors/arm-based-processors/overview.html

[14] Arrow Board and Box-Level Solutions for Arm Processors: https://www.arrow.com/

[15] ARM Cortex-A72 Processor Product Brief: https://developer.arm.com/Processors/Cortex-A72

[16] Embedded Computing Design article: "ARM's Cortex-A72 sets new performance standards for embedded systems" https://www.embedded-computing.com/processing/arms-cortex-a72-sets-new-performance-standards-for-embedded-systems

[17] Guidelines for Securing the Internet of Things — ENISA (europa.eu): https://www.enisa.europa.eu/publications/guidelines-for-securing-the-internet-of-things

[18] https://www.iiconsortium.org/pdf/IIC_PUB_G4_V1.00_PB-3.pdf

[19] Github.com, OWASP/IoT-Security-Verification-Standard-ISVS, https://github.com/OWASP/IoT-Security-Verification-Standard-ISVS/blob/master/en/images/ISVS-Overview-small.png, retrieved 2023-03-13

[20] GSMA, IoT Security Guidelines, https://www.gsma.com/iot/iot-security-guidelines-overview-document/, retrieved 2023-03-14

[21] The IoT Security Assurance Framework: https://www.iotsecurityfoundation.org/wp-content/uploads/2021/11/IoTSF-IoT-Security-Assurance-Framework-Release-3.0-Nov-2021-1.pdf

[22] Radio Equipment Directive (RED): https://single-market-economy.ec.europa.eu/sectors/electrical-and-electronic-engineering-industries-eei/radio-equipment-directive-red_en

[23] General Data Protection Regulation: https://gdpr.eu/what-is-gdpr/

[24] Network and Information Security (NIS) Directive: https://www.enisa.europa.eu/topics/cybersecurity-policy/nis-directive-new

[25] Cybersecurity Act: https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32019R0881

[26] The Medical Device Regulation (MDR): https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32017R0745

[27] The Cyber Resilience Act: https://digital-strategy.ec.europa.eu/en/library/cyber-resilience-act

[28] European Telecommunications Standards Institute (ETSI): https://www.etsi.org

[29] "D1.4 Use Cases Definition Final Version" Deliverable of the CROSSCON project.

[30]"D1.2 Requirements Elicitation Initial Technical Version" Deliverable of the CROSSCON project.

[31] "CROSSCON-Requirements-Table-v1.0 (with backlog) FOURTH Review Meeting" Excell file for requirements review and validation process.