Hardware Threat Landscape and Good Practice Guide

FINAL

VERSION 1.0

OPSEC

JANUARY 2017

www.enisa.europa.eu European Union Agency For Network And Information Security

About ENISA

The European Union Agency for Network and Information Security (ENISA) is a centre of network and information security expertise for the EU, its member states, the private sector and Europe's citizens. ENISA works with these groups to develop advice and recommendations on good practice in information security. It assists EU member states in implementing relevant EU legislation and works to improve the resilience of Europe's critical information infrastructure and networks. ENISA seeks to enhance existing expertise in EU member states by supporting the development of cross-border communities committed to improving network and information security throughout the EU. More information about ENISA and its work can be found at www.enisa.europa.eu.

Contact

For contacting the authors please use enisa.threrat.info@enisa.europa.eu
For media enquiries about this paper, please use press@enisa.europa.eu.

Acknowledgements

This study has been carried out in collaboration a group of experts in the area of hardware security, namely: Claire Vishik (Intel), David Oswald (University of Birmingham), Georg Sigl (Technical University of Munich), Sergey Bratus (Darthmouth College), Dimitris Pendarakis (IBM), Julien Touzeau (Airbus) and Wolfgang Klasen (Siemens). The group has provided valuable input, has supported the ENISA threat analysis and has reviewed ENISA material. Their support is highly appreciated and has definitely contributed to the quality of the material presented in this report.

Table of Contents

Executive Summary

IT systems have been part of everyday life since many years. However, over the last 5 years, mobile, embedded, and personal computing devices have become even more ubiquitous and even essential to many aspects of life. Examples for such devices are smartphones, tablets, car Multi Media Interfaces (MMIs), connected medical devices, home or enterprise access control/alarm systems, home automation, or also industrial control systems.

These classes of devices (herein referred to as mobile/embedded/personal computing devices) are either physically exposed in a particular way[1], lack security features of common IT systems[2], or both. The goal of this report is the compilation of a comprehensive landscape of hardware-related assets, threats, and good practices. This landscape provides basic information for manufacturers and developers who want to understand which threats their products are exposed to. Moreover it supports end-users who want to understand security aspects related to future products/technologies. Finally this report provides guidance on how existing good practices for the design, development, and implementation of embedded, mobile, and/or personal computing devices can contribute in providing protection.

The identified good practices were mapped to the developed threat landscape to identify possible gaps. This analysis showed that comprehensive good practices and security measures for hardware-related assets are available, however, not implemented widely enough. Several specific security controls or practices were identified and documented in section 8 to further complement the available good practices and close the identified gaps.

The most relevant identified gaps are:

- Lack of comprehensive and continuous use of platform security mechanisms by system developers as well as the integration of those by platform developers.
- Focus of good practices/available research on Bios/CPU firmware: Other firmware assets (such as chipsets or NICs) are barely covered by existing research and good practices.
- Lack of tamper detection: Detecting modifications of firmware is inherently hard as the tamper detection potentially has to rely on functionality offered by the firmware.

In addition to the above gaps, the following recommendations have been made:

- **Integration of threat analysis:** System developers must integrate threat analysis aspects into every step of the development. The good practice *Secure Embedded Design and Development Lifecycle* must take hardware-specific aspects, such as trust boundaries within a single system, into account.
- **Language security aspects:** During development, language security aspects should be taken into account. It should be evaluated whether languages such as Go or Rust can be used instead of C/C++ which are more prone to the introduction of memory corruption vulnerabilities.

[1] Such as exposing connector ports without the need for tampering or being mounted in publicly accessible areas.
[2] E.g. due to resource constraints.

- **Use and contribute to standards:** While there are efforts to provide industry guidance on the security of hardware-related assets, those efforts show certain gaps and need to be progressed to cover all aspects of secure development. Thus the existing standards should be used for the development and at the same time, gaps should be documented and closed by providing feedback and input for the standards.

Finally, interested readers will be in the position to deepen into aspects of threats, vulnerabilities but also mitigation by means of good practices taken into account and other comprehensive resources found on this subject.

1. Introduction

The aim of this document is the development of a threat landscape for attacks targeting firmware, embedded software, and hardware with a focus on modifications with malicious intent of mobile/embedded computing devices.

There have been various news reports where computing hardware was modified to monitor people – e.g. their location, personal data, or communications. Such modifications were carried out in the context of government/industry espionage [1] as well as motivated by personal interests (such as monetary advantages [2] or distrust). Even though the most known cases resulted from surveillance/monitoring attempts on persons, monitoring is by far not the only relevant threat scenario when assessing hardware-related attacks, which will be described during the threat analysis. This document will cover attacks/threats with the following characteristics:

- Non-invasive, i.e. such that do not result in permanent changes to the device;
- One of the following aspects applies:
    - Existing hardware is modified/extended;
    - Firmware of the device or one of its modules has been modified and
    - A vulnerability in the firmware of the device or one of its modules is exploited.

Section **Error! Reference source not found.** describes the scope of this document in more detail.

This document describes the various assets that can be affected by hardware-related attacks, the corresponding different possible attack vectors, weaknesses in current and common mobile/embedded computing platforms and potential countermeasures (also in the context of the hardware engineering process). While we strive for creating results which are applicable for computing environments in general (and thus also covering computing devices as diverse as vehicle MMIs, networked lawn mowers, or medical devices), certain threats, assets, or good practices will use specific examples (e.g. based on notebook computers or smartphones) which, however, can be adapted for other types of devices with similar architectures.

1.1 Scope

The focus of this document is on attacks against mobile/embedded computing devices which comply with certain characteristics. Before defining those characteristics, the computing devices in scope must be properly defined:

- **Embedded computing device** (or short: embedded device): "*An embedded device is a microprocessor-based system that is built to control a function or range of functions and is not designed to be programmed by the end user in the same way a PC is*" [80]. This definition indicates that embedded devices are often implemented on non-PC platforms, which also results in changes in the available computing resources – often those are constrained in some way(s). The definition also shows the modification of or insight into embedded devices is more difficult to achieve than for PC platforms.

- **Mobile computing device** (or short: mobile device): For this class of devices, we will use the intuitive definition of computing devices which are supposed/designed to be mobile, such as smart phones, notebook computers, or tablet computers.

Devices can of course be both mobile and embedded – smartphones are only one example for such a class of devices. Further examples for devices of one or both categories are personal and mobile computers, electronic consumer devices with connectivity (set-top box or TV, digital camcorder or camera), and IoT devices like smart meters, kitchen equipment or home automation systems.

The following list provides a more granular definition of the threats and attack vectors in scope of this document:

- **Non-invasive:** The attack does not result in permanent changes to the device. For example, the connection of a plug to an internal (such as PCIe) or external (such as FireWire) interface complies to this definition, however, the soldering of an additional chip onto existing soldering points or the extraction of communication bus circuit paths from a closed chip case does not. (The use of the existing soldering points with specific pliers however would not be invasive).

- One of the following aspects does apply:

    - **Modification/Extension of existing hardware:** The attack extends or modifies existing hardware. A good example for such a modification/extension is the Cottonmouth-1, which fits into an ordinary USB cable plug and is described in the NSA ANT Catalog [3] (refer also to Figure 1). It extends an existing USB cable in a non-invasive way and supports over-the-air attacks. This option allows the installation of Trojans on the target system which is connected to the modified USB cable. Other examples are devices which are plugged into existing internal or external interfaces (such as a FireWire plug or a PCIe device); those are likely to be more obvious.
    - **Firmware modification:** Firmware of the device/one of its module is modified via available mechanisms for modification (e.g. unauthenticated local update functionality, refer to Section 4.2 for more details). One example is the modification of the System Management Mode of modern CPUs, which allows the implantation of backdoors which cannot be detected by the operating system [5]. Adding firmware to the scope is particularly relevant as firmware often bears the potential to have as much impact on the computing device as the hardware itself (e.g. firmware typically has direct access to all hardware functionality).
    - **Firmware exploitation:** A vulnerability in the firmware of the device/one of its module is exploited, either locally or remotely. For example, [42] describes the remote exploitation of an Ethernet network interface card via traditional software vulnerabilities (which also exist in Firmware).

Explicitly not in scope of this document are:

- Attacks or analyses targeting the “silicon-layer” of the hardware, such decapping (e.g. using acid), thermal/interference measurement, or de-soldering of components.

- "Bugs" for eavesdropping the environment (audio and visual). Such bugs do extend the existing hardware, but typically only by means of using the power supply and not by interacting with the computing environment.

- Specific aspects of pure machine-to-machine communication/environments.

- Aspects of supply-chain security, in particular threats that have been incorporated in the hardware development environment and have thus led to backdoors that are implemented in the chip design.

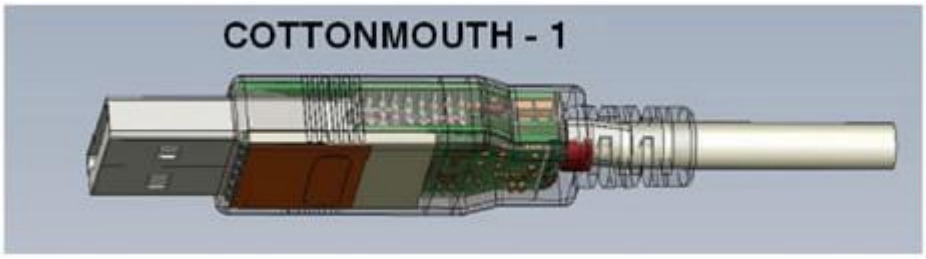

Figure 1: COTTONMOUTH-1 [Picture Source]

1.2 Target Audience

This thematic threat landscape can be used by developers, vendors/manufacturers, and customers/end users. For developers, it supports the process of threat analysis, evaluation of security controls, and definition of development practices while it raises awareness for the need to do so on the decision making level of vendors/manufacturers.

The report also describes relevant threat scenarios for customers and end users which need to be taken into account when considering the introduction of new hardware in existing/available components. In addition, it can also provide input for the development of security criteria for the selection process of new hardware/technologies.

The performed gap analysis provides input for researcher/research bodies/communities for the steering of research efforts on the areas of the identified gaps in good practices.

1.3 Structure

The remainder of this document is structured as follows:

- Chapter 2 describes the used methodology for the performed threat and asset analysis as well as the good practice and gap identification.
- Chapter 3 describes the various assets which are on different levels related to hardware or embedded/mobile/personal computing devices.
- Chapter 4 presents generic infrastructure threats which apply to hardware assets as well and, more importantly, threats which are very specific to hardware. The threats are also mapped to the different assets and complemented by the different impact they can have.
- Chapter 5 describes which threat agents are likely to execute/result in which threats based on the characteristics and capabilities developed over the different general ENISA Threat Landscapes.
- Chapter 6 describes the results of the research on available good practices for the security of hardware and embedded/mobile/personal computing devices.
- Chapter 7 lists the identified gaps in available good practices and their application/effectivity for the mitigation of the described threats.

- Chapter 8 gives recommendations to both apply the identified good practices and attempt to close the identified gaps.
- Chapter 9 provides summarizing remarks on the overall report.

2. Method

The methodology used in this threat landscape is in line with the methodology introduced (and thoroughly described) in the ENISA's Cyber Threat Landscape [41]. The following figure illustrates the terms used in this document and their relationships between each other:

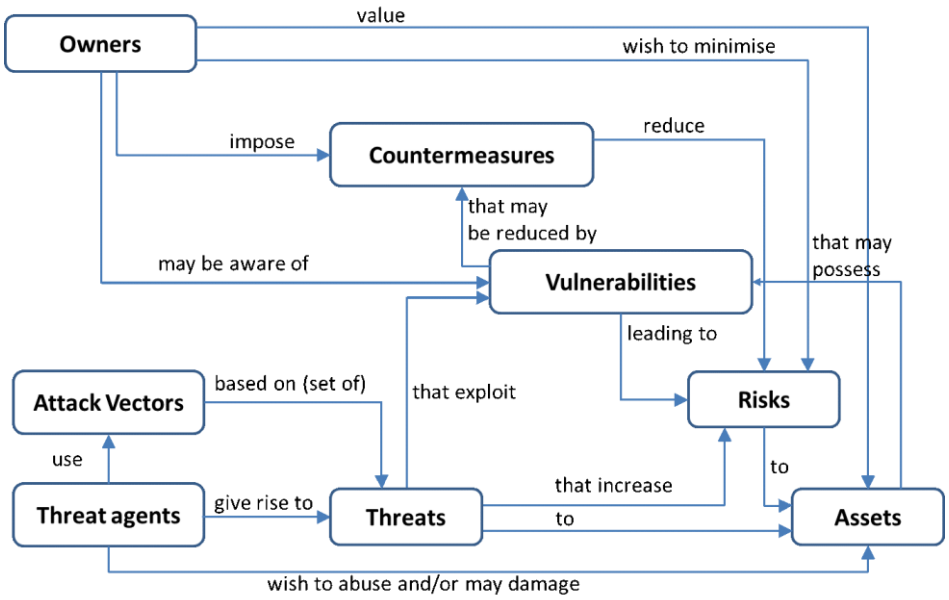

Figure 2: Threat-related Terms & Relations According to ENISA Threat Landscape 2013

The most relevant terms for this document are assets, threats, and countermeasures. In a first step, we list relevant hardware-related *threats* that can, if successfully materialized, have impact on *assets*. The threat analysis has been performed (according to ISO 27005:2011, 8.2.3) taking into account known incidents/attacks, information provided by asset owners/users/external experts/stakeholders and external threat catalogues. Assets have been identified in a comparable way. Where necessary, additional threats not originating from external resources (e.g. because no catalogues on firmware-specific threats or assets are available) have been developed to provide a complete view on the existing landscape of threats and assets.

Using this understanding of threats and assets, good practices were identified from existing recommendations, standards, and publications for the secure development/design of hardware assets. The identified good practices were mapped to the threat landscape. Wherever existing good practices did not suffice to mitigate a threat, a gap was identified and documented in the gap analysis. This information constitutes the recommendations for further future improvement of the hardware threat and good practice landscape.

2.1 Terms and Definitions

The following table lists and describes relevant terms used on a regular basis through this document.

TERM/ABBREVIATION	DEFINITION	SOURCE, IF REQUIRED
Asset	Anything that has value to asset owner (i.e. organization, external contractor, end-user, etc.)	ISO 27000:2009
Threat	Potential cause of an unwanted incident, which may result in harm to a system or organization	ISO 27000:2009
FW	Firmware, “equipment used for a particular purpose; especially: computer equipment”, also: Section 3.8.	Merriam-Webster
HW	Hardware	

Table 1: Definitions of used terms

3. Assets

As initially described, the scope of this document is the threat landscape analysis of hardware modifications of the personal/mobile/embedded computing devices. These types of devices embrace smartphones, tables, car MMIs, smart home devices, or also medical devices. The variety of devices also results in a variety of assets and different types of assets. To illustrate this using the example of a smartphone: The (integrity of the) device itself is a physical asset of financial (and potentially even emotional) value to users, the integrity of parts of the device (such as the battery) can affect the asset "user health" [6], and the information on the device consists of several other assets.

To provide a more structured landscape of assets, we use the following asset categories for the overall asset landscape, whereas the colours mentioned correspond to their illustration in **Error! Reference source not found.**:

- In the first category assets (highlighted in orange), the harm resulting from a threat can directly be related to the violation of a security requirement of one of those assets.
- In the second category assets (highlighted in green), logical or physical operational aspects and processes are included which, if impacted, can have an intrinsic negative impact as well as result in impact on first category assets.
- The third category assets (highlighted in blue) can result in impact on first and second category assets and can have an intrinsic negative impact if affected. These assets are also technical components and will be the relevant entities for the development of good practices.

For example, in **Error! Reference source not found.** the first category asset *User Health: Safety Against Malfunction* stands for the health of users of hardware. If the safety against malfunction is impacted, this can result in harm to the user. The third category asset *Battery Firmware* has intrinsic security requirements by itself (integrity of the hardware) and, if impacted by a potential threat *Malicious Firmware Modification*, can also impact the user health by developing high levels of heat or also irreversible destroy the device.

The following sections describe aspects of the different assets which are of particular relevance (e.g. for the remainder of the document and/or the better elaboration on the used methodology). The graphical presentation of these assets and their categories are as in **Error! Reference source not found.**.

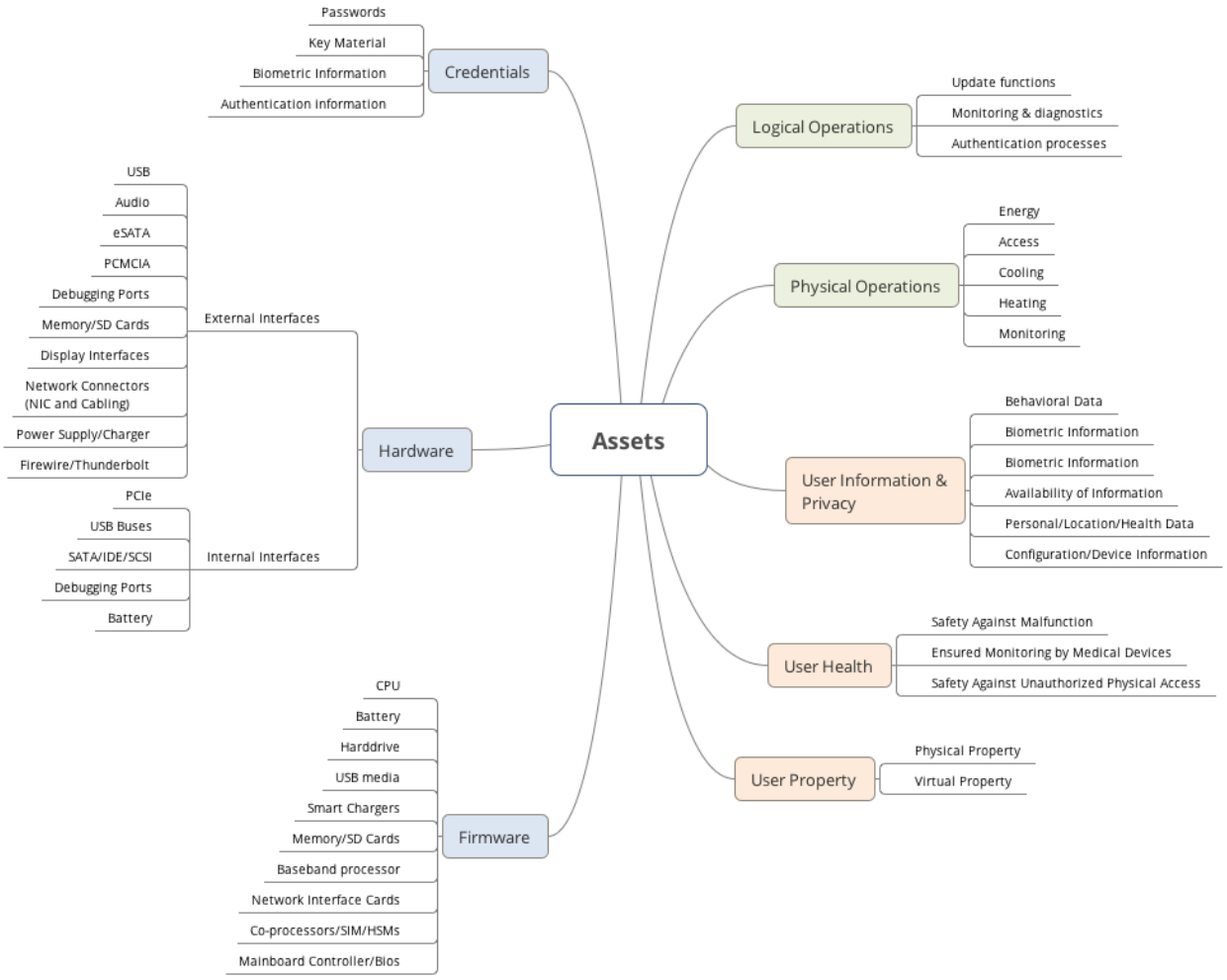

Figure 3: Asset inventory

3.1 User Property

User property refers to assets belonging to users which can be physical objects (such as electronics and other IT-items stored in a house or company building) and virtual objects (such as user data stored in these physical hardware objects). Loss of hardware will impact both types of user property; the hardware in a direct, immediate way but also virtual objects stored in the hardware. This class of assets illustrates the connection between digital threats and physical assets (which is often described by using the attribute cyber-physical) as well as the vanishing isolation between those when it comes to negative impact. The same holds true for the asset User Health (see below).

3.2 User Health

The health of hardware users is a very relevant asset and can be impacted in various ways: Batteries of devices can explode (which can also be achieved in a malicious, non-physical way), medical devices can fail/be modified to fail, and harm can come to users when access controls (such as digital door locks or security systems) fail. Human health is an asset of highest importance in any type of risk analysis and has traditionally not often been affected by IT-related threats.

3.3 User Information & Privacy

User information and privacy is less exclusively related to hardware-based threats than other asset groups, but specific information such as location data or audio/visual data is often tied more closely to the usage profiles of certain classes of mobile/personal/embedded devices. Various types and large amounts of information is processed on hardware assets where the violation of security objectives would have heavy impact on the users.

3.4 Credentials

Key material has similar characteristics as assets in the asset group user privacy, however, key material is also often related to the device itself and not only to the user. The variety of devices in scope results in all types of credentials which are stored or processed on them, such as passwords and other key material. In addition, the device classes in scope often serve as an additional factor in multi-factor authentication mechanisms.

3.5 Logical Operations

Logical operation mechanisms and processes, such as authentication, update, and monitoring & diagnostics, can be impacted by hardware implants. Authentication mechanisms can by bypassed, update procedures used to spread compromise to other devices, and monitoring & diagnostics functions can be (ab-) used to implant surveillance/monitoring functionality.

3.6 Physical Operations

Comparable to logical operations, physical operational functions such as cooling, heating, and energy can be impacted and result in negative harm to physical environments. At the same time, monitoring and access functionality can be negatively impacted to allow impact to other assets, e.g. physical assets which are not properly monitored/access protected any more.

3.7 Hardware

Hardware assets comprise various types of external and internal interfaces which can be affected by modification/extension as described in Section 1.1. External and internal refers to the typical casing of devices: If an interface is accessible without opening/tampering with the casing (such as typical USB ports), it is referred to as an external interface. Internal interfaces are accessible after opening/removing the casing but without modifying the hardware inside. For example, pins that expose a JTAG interface[3] are referred to as an internal interface, soldering points which would allow the connection of a JTAG interface are out of scope of this threat analysis. Furthermore, any interfaces/modifications which are only accessible/possible when physically modifying chips, circuit paths, cabling, or similar internal hardware parts are out of scope as well.

Hardware as an asset group can be impacted by threats in a twofold way: The hardware itself is a physical asset to users (based on value and function) which can be impacted, at the same time, hardware can be modified to impact other asset types, such as user health and property.

3.8 Firmware

Merriam-Webster defines firmware as "*computer programs that are contained permanently in a device*". While this general non-IT source may result in a too generic definition of firmware, various IT/academic sources also refer to firmware in the described way.[4] Having firmware update mechanisms (whether

[3] Refer e.g. to http://www2.lauterbach.com/pdf/training_jtag.pdf
[4] Refer e.g. to [FWCourse], where even Wikipedia is quoted for the definition of firmware.

intentionally developed by a vendor or as a malicious attack exploiting a vulnerability) in mind, this definition may be too strict for many types of software that are typically referred to as firmware. However, the idea that firmware is contained permanently in a device indicates the following characteristics which are relevant for this document:

- Firmware is contained in a device; this indicates a tight coupling and integration and in turn also means that firmware has control over hardware on a low level (which, as the next bullet item, will also be taken into account for the development of the threat landscape in Section 4.2)
- Firmware is associated permanently in a device; this indicates a lack of control of system software/operating systems over firmware. Firmware thus often is not directly visible to/accessible by typical system software/operating systems.

This definition would match well on firmware, for example of graphic cards or CPUs. However, operating system bundles on embedded devices, such as home routers or GPS devices, are also often referred to as firmware. For this document, both types of firmware are relevant and the aspect that firmware cannot be directly accessed from typical user interfaces is used to illustrate the lack of control over firmware.

Firmware modifications are a powerful way to impact all previous asset groups: Logical modifications can impact all functionality of and access to the previously mentioned assets, in addition, it is very hard to detect if implanted into certain devices.

4. Threats

The ubiquity of computing environments in all areas of life results in a very high need for security, safety, privacy, and resilience for the involved devices and services. The following subsections will describe the landscape for hardware-related threats.

4.1 Taxonomy of Hardware-related Threats

Based on the ENISA Threat Landscape 2015 (herein short: ETL15, [41]), the general categories of threats for the thematic Hardware Threat Landscape are:

- Nefarious activity/abuse (NAA): This threat category comprises intended actions that target IT systems with the purpose to steal/modify/tamper with/destroy assets
- Eavesdropping/Interception/Hijacking (EIH): This threat category comprises actions striving to access communication in an unauthorized way.
- Physical attacks (PA): This threat category comprises actions which aim to destroy, expose, alter, disable, steal or gain unauthorised access to physical assets as defined above.
- Damage (DAM): This threat category comprises intended actions which result in destruction, harm, or injury of property or health and can result in a loss of value/function.
- Unintentional Damage (UD): This threat category is comparable to DAM, however the impact is the result of an unintentional action.
- Failures or malfunctions (FM): This threat category comprises unwanted behaviour of an IT system affecting the ability to execute the desired function.
- Outages (OUT): This threat category comprises events leading to unexpected/undesired disruptions in the delivery (quality) of services – which are not limited to IT services.
- Disaster (DIS): This threat category is defined as serious disruption of the functioning due to some physical or man-made disaster.
- Legal (LEG): This threat category comprises legal actions of third parties with the potential effect to impact assets in various ways.

An overview of the assumed threats can be found in Figure 4.

In addition to the above general taxonomy, we also categorise threats depending on whether they (can) have specific Hardware-related aspects and generic threats to IT infrastructure systems. The generic threats have a broader scope and do not only apply to systems in the scope of this document but are still relevant as they can influence directly or collaterally the functioning of the devices. Several specific hardware-related threats can be specializations of these generic threats.

4.2 Hardware-/Firmware-specific Threats

In the following, we present types of threats that are specific to Hardware. Such threats may relate to different hardware-related assets, exploit vulnerabilities which are specific to the hardware assets in scope of this document, or require different handling when compared to traditional IT security approaches. The

described threats can cause/affect or be related to other threats in various ways. This way of specifying threats would not be suitable to determine the *most relevant* hardware-related *risks* due to its ambiguity (i.e. the successful manifestation of one threat can cause more threats to successfully manifest). However, listing threats in a more detailed (yet unfortunately ambiguous to a certain degree) manner ensures that this document provides guidance for readers with different backgrounds and expectations who do have to put less effort into understanding which further events could be caused by few accurate threats. In addition, it supports the design of good practices on multiple levels and taking potential security measures for multiple assets into account.

For each threat, sources are listed to provide background information or specific examples for the threat. This list is not exhaustive but meant to provide further background information and motivation for the relevance of the threat, where applicable. Several threats can inherently be derived from hardware aspects and thus do not require specific sources as more technically sophisticated threats might do.

Hardware Modification: The modification of hardware can be performed in various ways; this threat focuses on non-intrusive ways (as described in Section 3.7) which (ab-) use available interfaces (such as Firewire, PCI Express, or USB) to modify hardware to carry out/support unintended functions. The threat table below will contain various examples of potential hardware modifications.

Relevant Sources: [1], [2], [3], [4], [7], [8], [9], [10], [11], [12], [13]

Firmware Modification: The modification of firmware is less intrusive than the physical modification of hardware and can have very similar effects. The function of the hardware can be modified, processed data intercepted, and security functionality be bypassed by modifying (i.e. exploiting weakness of) the logic which manages the hardware. Firmware modifications can be implanted in different ways: Using existing firmware update mechanisms, exploiting a vulnerability in the firmware already loaded onto the device, using binary firmware loading mechanisms[5], or exploiting the lack of access control/write protection of firmware storage (e.g. unlocked NVRAM during boot). [57] provides a comprehensive list of possibilities to update firmware (in both authorized or unauthorized way).

Relevant Sources: [5], [32], [34], [36], [37], [56], [57], [72], [85]

Remote Firmware Attacks: Attacks which can compromise the firmware of a device in a remote way (as for example demonstrated in [42] where software vulnerabilities are exploited in the firmware of an Ethernet network interface card) result in the same impact as described in Firmware Modification above, however, no logical or physical access to the device is required. If the attack is carried out in a sophisticated way (e.g. by immediately modifying essential functions), there is also no way for traditional security controls to detect the attack.

Relevant Sources: [37], [49], [52]

Attack Persistence: Traditional security controls focus on the prevention and detection of logical threats on the application or operating system level. Attacks that are carried out in a way that bypasses those levels (e.g. by attacking firmware which may not even be accessible by the operating system/application or modifying the functioning of hardware in a transparent way) cannot be detected by traditional controls or mechanisms to verify the integrity of the computing environment. This results in a very high level of attack

[5] A relevant number of devices only offers basic functionality to load firmware which is provided by the operating system which then provides the actual functionality of the device. This mechanism is referred to as binary firmware loading.

persistence that can be achieved by attackers and cannot even be countered with a complete system re-install.

Relevant Sources: [1], [2], [3], [4], [7], [8], [9], [10], [11], [12], [13], [5], [34], [36], [37]

Traffic Sniffing: The access to network traffic is a common threat in typical IT environments. However, in the context of hardware-related attacks, traffic sniffing is not limited to network connections but can also be carried out on internal buses and connections, such as the memory or hard drive bus. Those bus systems traditionally do not assume threats from within those system/devices which are physically connected so that no compensating controls are implemented.[6]

Relevant Sources: [28], [29]

Surveillance: Surveillance is a specific type of access to information that combines the basic information access with a focus on personal/private data and the use of hardware to gather information from the physical world, for example by (ab-) using microphones, cameras, or location data. Typical personal mobile computing environment comprise various sensors that can be abused to form strong surveillance capabilities.

Relevant Sources: [50], [85], [2]

Data Tampering/Spoofing: Comparable to surveillance threats, the tampering or spoofing of data on mobile computing devices can have wider impact than typical data tampering: Spoofed location, audio, or visual data can lead to a variety of abuse scenarios.

Relevant Sources: [2], [85]

Information Access: Mobile computing devices store all types of information which often form/represent significant parts of the identity and belongings of users. Hardware-related attacks can lead to a completeness of information access that extends the capabilities of typical logical IT threats and thus need to be covered in a dedicated manner.

Relevant Sources: [2], [85]

Malfunction: In a connected world that is supported by computing devices in all areas of life, the malfunction of devices can result in a variety of harm and negative impact. Several specific threat scenarios are for example the malfunctioning of medical devices (performing critical tasks on a patient), access control systems (preventing unauthorized access to people's homes), or monitoring systems (e.g. for hazards such as fire).

Relevant Sources: [6], [83]

Denial-of-Service: Comparable to malfunction, (successful) denial-of-service attacks are comparable to maliciously induced malfunction. This threat represents the denial-of-service *of* mobile/personal/embedded devices, e.g. the crash of a smartphone, the outage of a monitoring solution, or the error state of an alarm system. Denial-of-service attacks *originating from* mobile/personal/embedded devices (e.g. as happened recently in the case of the Mirai Botnet [81]) can be

[6] There are new technologies like Intel® SGX [43] which change this landscape with regard to certain threats, however, this will be covered in Section 6.

a threat for the same classes of devices, however, it would be a generic threat. In addition, it is also a potential effect/impact of a successful materialization of a threat like *Remote Firmware Attacks*.

Relevant Sources: [81], [82], [83], [84]

Modification-of-Service: Mobile computing devices provide a variety of services, tampering with the way the service is delivered or changing the result/outcome of the service delivery, various specific threat scenarios can be realized. While the malfunctioning of a device can impact various assets, the modification-of-service can in addition pave the way for further threats/attacks.

Relevant Sources: [86]

Property Losses: Attacks against hardware can lead to access to both physical (e.g. items in an apartment) and logical (e.g. online financial funds) property. The specific aspect to be assessed is the possibility to also attack non-networked devices such as smart locks or access control systems.

Relevant Sources: [87]

Destruction of Hardware: This threat is a specification of typical damage-related threats. For mobile computing devices, hardware damage can be (maliciously) induced via logical attacks, e.g. by tampering with battery firmware.

Relevant Sources: [44], [84]

Loss of Compliance: Mobile computing devices are used in various areas, some of those requiring strict certification (e.g. FDA approval or the CE marking) for any computing device to be used. Modification of those devices in any way can result in a loss of certification and thus compliance violations. Tampered devices can also violate regular security violations when it comes to access control requirements.

Relevant Sources: [88]

Waste of Resources: Attacks on certain types of mobile computing devices can result in a waste of resources. While energy can also be wasted as a result of logical attacks, even bigger amounts and different types of resources (e.g. water) can be wasted when control systems are attacked.

Relevant Sources: [89]

4.3 Generic Infrastructure Threats

IT systems beyond personal computing devices (such as private smartphones) are almost always part of a bigger system (e.g. providing services in a networked application landscape or performing access control tasks to a building). The resulting interdependencies introduce additional generic infrastructure threats. Though not specific to hardware, such threats may cause impact to hardware through damages in software (e.g. affecting software that is embedded in the hardware). Sources such as the ENISA annual Threat Landscape and ISO 27005 have been taken into account for the assessment of generic threats which also affect the entities in scope of this document.

4.4 List of Hardware-related Threats

In the following, we present two tables of Hardware-related threats. These tables list Hardware-specific threats (**Error! Reference source not found.**), and generic network threats (**Error! Reference source not found.**), respectively, structured according to the threat taxonomy described above. For each threat, the tables provide:

- A brief description of the threat in the column "Threats". In cases where a threat falls under more than one category in the ENISA taxonomy, the additional categories are identified in the description of the threat.
- The assets that the threat potentially affects (see column "Asset Types"). This description refers to the asset listing produced earlier as part of the project.
- The potential effect of the threat described in terms of the basic security properties that a threat can compromise, i.e., confidentiality, integrity or availability (see column "Potential Effects").

THREAT TYPES	THREAT	POTENTIAL EFFECTS	ASSET TYPES
Nefarious Activity/ Abuse	Firmware Modification, e.g. of CPU, internal/external Controllers (e.g. hard drive/USB media), chipsets, smart chargers, smart batteries, co-processors, NICs • Exploiting firmware vulnerabilities • Abusing update functionality • Abusing binary firmware loading mechanisms	• Information integrity • Information confidentiality • Information destruction • Software asset integrity • Service availability • Service functionality • Outage	• Logical Operations • Physical Operations • Hardware • Firmware
	Remote firmware attacks, e.g. in network interface cards • Memory Corruption Vulnerabilities • Logical Flaws • Backdoor Functionality • Remote management functionality (e.g. [45])	• Information integrity • Information confidentiality • Information destruction • Software asset integrity • Service availability • Service functionality • Outage	• Logical Operations • Physical Operations • Hardware • Firmware

	Attack Persistence • Firmware modification/Bootkit	• Information integrity • Software asset integrity	• Logical Operations • Physical Operations • Hardware • Firmware
	Information Access (Can also be *Physical Attacks*)	• Information confidentiality	• User Property • User Information & Privacy • Logical Operations
Eavesdropping/ Interception/ Hijacking	Traffic Sniffing • Network level • Internal Bus level • Memory level	• Information integrity • Information confidentiality • Information destruction • Software asset integrity	• User Information & Privacy • Logical Operations • Physical Operations
	Surveillance of… • Location • Audio • Visual data • Behaviour	• Information confidentiality	• User Property • User Information & Privacy
	Data Tampering/Spoofing of… • Location • Behaviour	• Information integrity • Information destruction	• User Information & Privacy

Physical Attacks	Hardware Modification
	• External Hardware Trojan
	  ○ Regular plug
	  ○ Transparent, with pass-through functionality
	• Internal Hardware Trojan
	• Temporary hardware access for system modification
		• Information integrity
		• Information confidentiality
		• Software asset integrity
		• Service functionality
			• Logical Operations
			• Physical Operations
			• Hardware
			• Firmware

	Property Losses • Access control bypass (e.g. smart lock) • Disabling of monitoring/alerting (e.g. alarm systems) • Unlock attack (e.g. in vehicles)	• Property availability • Property destruction	• User Property
Damage	Destruction of Hardware • Overheating • Explosion • "Bricking", e.g. destruction of firmware • Disabling of interfaces	• Property availability • Property destruction • User harm	• User Property • User Health • Physical Operations • Hardware
	Waste/destruction of Resources • Excessive Heating/use of heat - producing resources • Excessive energy consumption • Excessive use of water/physical resources controlled by a computing control system	• Property availability • Property destruction • Environment harm	• User Property • Physical Operations
Failures or malfunctions	Malfunction • Failure of medical devices • Overheating/explosion of batteries • Failure of control/production systems • Failure of access systems • Failure of alarm systems • Outage of monitoring system	• Service availability • Outage • Property availability • User harm	• User Property • User Health • Physical Operations • Hardware
	Modification-of-Service • Wrong treatment by medical devices • False negative reporting by alarm/monitoring systems	• Property availability • User harm	• User Property • User Health • User Information & Privacy • Logical Operations • Physical Operations

	• Granted access for unauthorized parties by access control systems		• Hardware • Firmware
Outages	Denial-of-Service • Flooding/volumetric attack • Software bug/exploit • Logical flaw	• Service availability • Outage • User harm	• User Property • User Health • Logical Operations • Physical Operations
Legal	Loss of Compliance • Voidance of certification/validation approvals • Violation of contractual requirements • Violation of internal/external compliance requirements • Violation of data protection laws	• Software asset integrity • Service availability • Reputation damage	• Logical Operations • Physical Operations

Table 2: Hardware-specific Threats

THREAT TYPES	THREATS	POTENTIAL EFFECTS	ASSET TYPES
Nefarious Activity/Abuse	Unauthorized use	Information integrity Information destruction Service availability Service functionality Outage	Physical Operations Hardware

	Abuse of rights	Information integrity Information confidentiality Information destruction Software asset integrity Service availability Service functionality	User Information & Privacy Logical Operations
Eavesdropping/ Interception/Hijacking	Physical eavesdropping/shouldersurfing	Information integrity	User Information & Privacy Logical Operations
	Lawful interception	Information confidentiality	User Information & Privacy
Physical Attacks	Fraud	Information confidentiality Property availability	User Property Hardware
	Sabotage	Service availability Service functionality Outage Property destruction User harm Reputation damage	Logical Operations Physical Operations Hardware
	Theft	Property availability	Hardware User Property
	Information Leakage/sharing	Information confidentiality	User Information & Privacy

	Unauthorized physical access/entry	Information integrity Information confidentiality Information destruction Software asset integrity Service availability Service functionality Outage Property availability Property destruction	User Information & Privacy Logical Operations Physical Operations Hardware User Property
Damage	Vandalism	Property destruction User harm	User Property User Health
	Terrorist Attack	Property destruction User harm	User Property User Health
Unintentional Damage	Misuse	Property destruction Service availability Service functionality Outage	User Property User Health
	Maintenance error	Property destruction Service availability Service functionality Outage	Logical Operations Physical Operations Hardware User Property

	Erroneous use	Property destruction Service availability Outage	Logical Operations Physical Operations Hardware User Property
Failures or malfunctions	Hardware failure	Property destruction Service availability Outage	Logical Operations Physical Operations Hardware User Property
	Software failure/bug	Service availability Outage	Logical Operations Physical Operations
Outages	Communication outage	Service availability Outage	Logical Operations Physical Operations
	Power outage	Service availability Outage	Logical Operations Physical Operations
Disaster	Natural Disasters	Service availability Outage Property destruction	User Property Logical Operations Physical Operations Hardware
	Fire	Service availability Outage Property destruction	User Property Logical Operations Physical Operations

			Hardware
Legal	Breach of SLAs	Financial losses	Logical Operations
	Breach of Legislation	Financial losses/legal actions	User Information & Privacy Logical Operations
	Abuse of personal data	Financial losses/legal actions	User Information & Privacy

Table 3: Generic Network Threats

5. Threat Agents

The ENISA Threat Landscape 2015 [41] refers to a threat agent as "characterizations of malicious actors (or adversaries) representing a cyber-attack threat including presumed intent and historically observed behaviour". Furthermore, [41] lists and describes the following categories of threat agents which will also be used in this document:

- Cyber Criminals
- Insiders (Employees)
- Online Social Hackers
- Cyber Spies (Nation States, Corporations)
- Hacktivists
- Cyber Fighters
- Cyber Terrorists
- Script Kiddies

The threat agent categories differ when it comes to motivation, capabilities and the possibilities, latency, and sources to gather information about the agents.

When it comes to hardware-related threat agents, the differentiation into threat agents with physical access to hardware (or the motivation and capabilities to establish physical access) and without physical access is relevant: Certain hardware-related threats only emerge from threat agents with physical access to hardware. Threat agents with (the means/motivation to establish) physical access can be distinguished into groups that inherently have access to systems (such as Insiders) or groups that have sufficient motivation and capabilities to also establish physical contact, such as cyber spies or criminals.

Various incidents (such as the physical access to an individual laptop computer [2] or the mass-development of attacks against embedded devices [49]) shows that attacks which require physical access are typically restricted to nation-state actors/agents or few highly funded and motivated cyber criminals.

For any risk management process, it is crucial for asset owners to be aware of which threats can emerge from which threat agent groups. The following table presents a mapping between the listed threat agents and the threat groups described in Section 4 (which can be related to threat agents) which may emerge from the threat agents:

	CYBER CRIMINALS	INSIDERS	ONLINE SOCIAL HACKERS	CYBER SPIES	HACKTIVISTS	CYBER FIGHTERS	CYBER TERRORISTS	SCRIPT KIDDIES
Nefarious activity/abuse	•	•	•	•	•	•	•	•
Eavesdropping/ Interception/Hijacking	•	•		•			•	•
Physical attacks		•		•				
Damage	•	•			•	•	•	•
Unintentional Damage	•	•			•	•	•	•
Failures or malfunctions	•	•			•		•	
Outages	•	•				•	•	•

Table 4: Involvement of various threat agent groups in cyber-threats

6. Good Practice of Hardware-related Security Measures

This section provides a review of existing controls, tools and practices for hardware-related threat mitigation that have been identified through literature research and discussion with an expert group convened by ENISA for this purpose.

The performed analysis of available publications considered established research programmes, conference papers/presentations, innovation projects, national/international/de-facto standards, and last, but not least, contributions of the security research community.

According to the scoping performed in Section 1.1, supply chain security aspects which are relevant for invasive/integrated modification of hardware was not in scope of the good practice research.

Furthermore, it must be highlighted that many traditional IT security controls also do apply to the systems/assets in scope of this document. For example, it is in general a good idea to implement the concept of *Data Avoidance* when designing systems, however, this control is not specific to hardware-related threats and how those can be mitigated. Thus the remaining section focusses on good practices that are only or especially relevant when mitigating hardware-related threats.

The following tables describe the identified good practices. The sources listed with each practice provide either or both a motivation for the practice or further details on the implementation/application of the practice.

Target Audience	Developers
Title	Minimal Hardware Access
Description	Physical access options to interfaces that provide access to sensitive device functionality (e.g. DMA capabilities or OS boot) should be removed as far as possible in the production system design. This applies in particular to interfaces which are often used for debugging in development (such as JTAG, SPI, or I^2C). If they cannot be enabled, they should be disabled logically/in software and/or make them harder to access (e.g. by using TSOP vs BGA chip beds Chips should be designed in a way that relevant communication buses do not run close to the outside/edges
Sources	[39], [53], [27]
ID	MinHWAccess

Target Audience	Developers, Vendors
Title	Lock Logical Access
Description	Logical access to sensitive system functions/storage should be restricted as much as possible. Hardware functionality to lock write access to relevant memory regions such as SMRAM should be used. Restrict access via interfaces like SPI, I^2C, or JTAG. Unnecessary boot options/order should be disabled.

	Platform functionality (e.g. CSM, SMM_BWP, BLE, BIOSWE, SPI Memory Protection, DENY_EXECUTE_ ON_SECURITY_VIOLATION /QUERY_USER_ ON_SECURITY_VIOLATION) must be correctly configured to not allow unauthorized access. Vendor documentation for the used hardware (such as the Intel handbooks) must be used to determine access restriction options. The CHIPSEC tool provides functionality to test/verify various platform security mechanisms. For hardware developers, the use of memory protection units should be evaluated.
Sources	[39], [33], [58], [60], [31], [27]
ID	LockLogicalAccess

Target Audience	Developers
Title	Secure Embedded Design and Development Lifecycle
Description	A secure embedded design and development lifecycle must be used for the development of hardware, firmware and mobile computing devices. The following aspects are of particular relevance beyond the aspects of typical secure development guidelines: • Secure coding guidelines must be specific for hardware-related development and languages. • Consider adding extra variable integrity validity checks on critical values to prevent “bricking” of systems should a value be improperly changed. • Rely on stable software components. Updates are often costlier than in traditional IT systems. • Rely on software components with long support times. • Trust boundaries must be reviewed: While typical trust/threat analysis often considers the local system trustworthy, this is not the case when assessing hardware-related threats. For example, firmware update mechanisms process data provided within the same system, however, the code processing a signed firmware upgrade bundle may be the only code processing data from other system components at all, making it an exposed interface. • Implement segregation of duties, least privileges, and different trust zones for different system services/functionality. Sources like [62]/[63] describe typical limits of embedded/hardware-related platforms, however, modern platforms offer sufficient resources to implement trust areas. • Only unmodified tools and components must be used for the development. This applies to both software and hardware and must be verified throughout the whole supply and development chain.
Sources	[39], [33], [60], [66], [67], [27], [78], [90]
ID	SDLC

Target Audience	Developers, Vendors
Title	Firmware Tamper Detection

Description	Modifications to system firmware should be detected by the system. Depending on the type of firmware tampering, detection is a very hard problem, however, certain mechanisms can and should be implemented: • Verification of timing anomalies which can be induced by interception/eavesdropping • Verification of state anomalies, e.g. content of certain HW-related ports/registers • Verification of the deployed BIOS with known-good sources • For very high protection need, the deployment of dedicated co-processors with the purpose of software integrity verification.
Sources	[27], [74], [75], [76], [77]
ID	FWTD

Target Audience	Developers
Title	Secure Update/Modification Management
Description	The system must support secure ways to handle modifications (e.g. by means of updates) to the firmware. Intentional modifications (e.g. a user installing an legitimate vendor update) must be possible, unauthorized modification must not be possible (such as an attacker trying to install a modified firmware using the official update mechanism – unofficial mechanisms are covered in LockLogicalAccess). The sources listed below contain detailed guidance, however, the following aspects should be particularly taken into account: • Updates should be signed in cryptographically secure way. Guidance on that can be found in NIST SP 800-89, NIST FIPS 186-3, or NIST SP 800-131A • The Root of Trust for Update (RTU) should be stored in a tamper-protected way, e.g. using hardware key stores. Those key stores must be properly closed after usage. • Standardized update mechanisms (such as the UEFI Update Capsule) should be considered during development. • The lifetime of the RTU must be adjusted to the expected lifetime of devices and update frequency, which is often much longer for embedded devices than regular IT devices. • Evaluate computing resources limits (such as limited storage when extracting update bundles) and environmental factors (such as limited network connectivity/connectivity with only specific protocols) to avoid update starvation or even bricking.
Sources	[39], [38], [58], [61], [37], [51], [27]
ID	SUM

Target Audience	Vendors, Industry
Title	Support Secure Development and Verification Standards
Description	Industry-wide secure development and verification standards should be supported and implemented. Compliance with development standards should be documented in an open and transparent way, the verification

	standards should result in publicly available documents covering the security posture of the asset.
Sources	[39], [67]
ID	SecStandards

Target Audience	Developers
Title	Open Security Validation
Description	Customers should be enabled to verify compliance on their own. Tools used in the course of SecStandards should be made publicly available to enable customer to verify test results in an independent way. Tools (e.g. for the discovery of testing key material) to ensure production readiness should be openly shared.
Sources	[38]
ID	OpenSecVal

Target Audience	Vendors, Industry
Title	Avoid Backdoor and/or PhoneHome Functionality
Description	Backdoor functionality must not be implemented. Various discoveries (e.g. [68]/[69]) show that those backdoors can and will be discovered and used for malicious purpose. Functionality used to connect back to vendor/manufacturer services (often referred to as phone home) should be evaluated for its necessity. If the functionality is required for the function of the device, it must be implemented taking highest security requirements for typical IT/software systems into account.
Sources	[38]
ID	BD-PhoneHome

Target Audience	Vendors, Industry
Title	User Awareness Process
Description	Vendors must create a process to inform end users about both security functionality, security incidents, and available security updates. Users must be made aware of potential inherent security problems with using the system or exposure that results from using the device (e.g. that data can be accessed when the device is lost or eavesdropping cannot be prevented due to very specific device use cases/design requirements).
Sources	[39], [38]
ID	UserAwareness

Target Audience	Vendors, Industry, Developers
Title	Secure Key Storage
Description	Devices should provide a secure key storage for key material. Various standards/de-facto standard solutions exist on key storage (e.g. TPM, SIM, SmartCards, or other HSMs). The secure key storage is also required by SUM.

Sources	[39], [27]
ID	SecKeyStor

Target Audience	Developers
Title	Platform Security Mechanisms
Description	For firmware/software developer: Available platform security mechanisms (such as DEP, ASLR, but also logical access restrictions described in LockLogicalAccess) must be used and correctly enabled/set, at least/the latest during the release process for the system. Access to memory regions should be restricted according to the system needs, e.g. using technologies like memory protection units or I/O MMUs (implemented e.g. by Intel's VT-d or AMD's Vi). For hardware/platform developer: Platform security mechanisms should be integrated, made available, and clearly documented to developers using the platform. A security best practices guide for the platform should be published.
Sources	[39], [27]
ID	PlatformSec

Target Audience	Developers
Title	Establish Chain of Trust
Description	It should be possible to establish a chain of trust from the initial hardware booting steps to the execution of the operating system. This relies on HSM and SUM, but also includes additional functionality in the boot loader loading process. Technologies like Intel TXT should be leveraged, if available by PlatformSec.
Background Information/ Sources	[39], [51]
ID	CoT

Target Audience	Developers
Title	Language Security
Description	C and C++ are the main languages when it comes to hardware-related development. However, those languages are also more prone to result in software vulnerabilities (mainly in the memory corruption area). Languages like Go and Rust also provide the option to create compiled executable code but also introduce type-safety, garbage collection, and other security-relevant characteristics. The use of such a language (or at least alternative programming paradigms as described in [70]) should be evaluated for hardware-related programming to prevent certain types of vulnerabilities in the first place.
Sources	[54], [55] (provides example for feasibility), the following two patents illustrate the need for the proposed approach: https://www.google.com/patents/US7761701

	https://www.google.com/patents/US7162626
ID	LangSec

Target Audience	Developers
Title	Secure by Default
Description	The security paradigm Secure by Default is not specific to hardware-related assets/embedded systems/mobile computing devices, however, comparable to PlatformSec, is often neglected on those systems for various reasons. Hence developers should ship releases with a secure default configuration, with particular regard to enforced authentication, supported strong authentication mechanisms, use of encryption features (see also Crypto) and reliable authorization components.
Sources	[62]/[63] illustrate limitations and deviations from security best practices when it comes to hardware-related functions/embedded devices, thus motivating the need to require strong defaults despite any limitations. [27]
ID	SecDefault

Target Audience	Developers
Title	Encryption of Data at Rest and in Transport
Description	Embedded or mobile computing devices often comprise less computing power than typical computing devices. This lack of resources was often used to argue for a lack of encryption since not enough computing power is available. However, sources like [64] and the improved cryptographic performance of elliptic curve-based cryptography (with particular regard to embedded devices, refer e.g. to [71]) show that the use of cryptography on modern embedded devices is feasible. Thus it must be ensured that secure transport and storage mechanisms are used wherever necessary, e.g. when it comes to wireless transport, pairing mechanisms, encryption of key material or user data.
Sources	[62]/[63] illustrate limitations and deviations from security best practices when it comes to hardware-related functions/embedded devices, thus motivating the need to require typical security controls despite any limitations. [64], [26], [27]
ID	Crypto

Target Audience	Developers
Title	Remote Wiping
Description	Mobile computing devices should comprise secure remote wiping features. While this good practice is in conflict with BD-PhoneHome, its use is mandated by international standards when it comes to mobile devices. Thus the security relevance of this good practice should be evaluated for the specific device use case and, if deemed necessary, implemented in a secure way integrating strong authentication and authorization mechanisms.

Sources	ISO 27002:2013, 6.2.1
ID	Wipe

6.1 Mapping of Good Practices to Threats

The following table presents a mapping of good practices to threats. This mapping allows to identify gaps in the available practices:

THREAT TYPES	THREAT	PARTIALLY ADDRESSED BY…
Nefarious Activity/ Abuse	Firmware Modification, e.g. of CPU, internal/external Controllers (e.g. hard drive/USB media), smart chargers, smart batteries, co-processors, NICs • Exploiting firmware vulnerabilities • Abusing update functionality • Abusing binary firmware loading mechanisms	• LockLogicalAccess • SDLC • SUM • SecStandards • OpenSecVal • PlatformSec • LangSec • UserAwareness
	Remote firmware attacks, e.g. in network interface cards • Memory Corruption Vulnerabilities • Logical Flaws • Backdoor Functionality • Remote management functionality (e.g. [45])	• SDLC • SecStandards • OpenSecVal • PlatformSec • BD-PhoneHome • LangSec • UserAwareness
	Attack Persistence • Firmware modification/Bootkit	• FWTD
	Information Access (Can also be *Physical Attacks*)	• Crypto • Wipe • SecKeyStor • UserAwareness
Eavesdropping/ Interception/ Hijacking	Traffic Sniffing • Network level • Internal Bus level • Memory level	• Crypto • SecDefault • UserAwareness • FWTD
	Surveillance of… • Location • Audio • Visual data • Behavior	• Crypto • Wipe • BD-PhoneHome • FWTD
	Data Tampering/Spoofing of… • Location • Behavior	• Crypto • Wipe • BD-PhoneHome • FWTD
Physical Attacks	Hardware Modification	• MinHWAccess

	• External Hardware Trojan
	  ○ Regular plug
	  ○ Transparent, with pass-through functionality
	• Internal Hardware Trojan
	• Temporary hardware access for system modification
		• CoT
		• LockLogicalAccess

	Property Losses • Access control bypass (e.g. smart lock) • Disabling of monitoring/alerting (e.g. alarm systems) • Unlock attack (e.g. in vehicles)	• Wipe • SecDefault • MinHWAccess • LockLogicalAccess • SDLC • OpenSecVal
Damage	Destruction of Hardware • Overheating • Explosion • "Bricking", e.g. destruction of firmware • Disabling of interfaces	• MinHWAccess • LockLogicalAccess
	Waste/destruction of Resources • Excessive Heating/use of heat - producing resources • Excessive energy consumption • Excessive use of water/physical resources controlled by a computing control system	• SecDefaults • MinHWAccess • LockLogicalAccess
Failures or malfunctions	Malfunction • Failure of medical devices • Overheating/explosion of batteries • Failure of control/production systems • Failure of access systems • Failure of alarm systems • Outage of monitoring system	• SDLC • SecStandards • OpenSecVal • PlatformSec • SecDefult
	Modification-of-Service • Wrong treatment by medical devices • False negative reporting by alarm/monitoring systems • Granted access for unauthorized parties by access control systems	• SDLC • SecStandards • OpenSecVal • PlatformSec • SecDefult
Outages	Denial-of-Service • Flooding/volumetric attack • Software bug/exploit	• SDLC

	• Logical flaw	
Legal	Loss of Compliance • Voidance of certification/validation approvals • Violation of contractual requirements • Violation of internal/external compliance requirements • Violation of data protection laws	• SDLC • SUM • SecStandards • OpenSecVal

Table 5: Mapping of good practices to threats

7. Gap Analysis

The following table lists assets which are not/only partially covered by the described good practices structured by the relevant threats:

THREAT TYPES	THREAT	PARTIALLY ADDRESSED BY…	GAPS
Nefarious Activity/ Abuse	Firmware Modification, e.g. of CPU, internal/external Controllers (e.g. hard drive/USB media), smart chargers, smart batteries, co-processors, NICs • Exploiting firmware vulnerabilities • Abusing update functionality • Abusing binary firmware loading mechanisms	• LockLogicalAccess • SDLC • SUM • SecStandards • OpenSecVal • PlatformSec • LangSec • UserAwareness	Assets not covered: • - Issues: • Vulnerability history shows that good practices are not applied globally/per default.
	Remote firmware attacks, e.g. in network interface cards • Memory Corruption Vulnerabilities • Logical Flaws • Backdoor Functionality • Remote management functionality (e.g. [45])	• SDLC • SecStandards • OpenSecVal • PlatformSec • BD-PhoneHome • LangSec • UserAwareness	Assets not covered: • - Issues: • Backdoors are still being discovered despite existing good practices.
	Attack Persistence • Firmware modification/Bootkit	• FWTD	Assets (partially) not covered: • Firmware Issues: • Existing hardware (architectures) do not offer functionality/interfaces for logical tamper detection.

			• Inherent problem of tamper detection when measurement data can be tampered with.
	Information Access (Can also be *Physical Attacks*)	• Crypto • Wipe • SecKeyStor • UserAwareness	Assets (partially) not covered: • - Issues: • -
Eavesdropping/ Interception/ Hijacking	Traffic Sniffing • Network level • Internal Bus level • Memory level	• Crypto • SecDefault • UserAwareness • FWTD	Assets (partially) not covered: • System-internal bus/data transfer Issues: • System architectures do only support limited internal encryption features.
	Surveillance of… • Location • Audio • Visual data • Behavior	• Crypto • Wipe • BD-PhoneHome • FWTD	Assets (partially) not covered: • - Issues: • -
	Data Tampering/Spoofing of… • Location • Behavior	• Crypto • Wipe • BD-PhoneHome • FWTD	Assets (partially) not covered: • - Issues: • -
Physical Attacks	Hardware Modification • External Hardware Trojan ○ Regular plug	• MinHWAccess • CoT • LockLogicalAccess	Assets (partially) not covered: • Hardware Issues:

	○ Transparent, with pass-through functionality • Internal Hardware Trojan • Temporary hardware access for system modification		• Depending on user awareness. • DMA functionality often required. • Differentiation authorized/unauthorized HW access difficult to measure.
	Property Losses • Access control bypass (e.g. smart lock) • Disabling of monitoring/alerting (e.g. alarm systems) • Unlock attack (e.g. in vehicles)	• Wipe • SecDefault • MinHWAccess • LockLogicalAccess • SDLC • OpenSecVal	Asset (partially)s not covered: • - Issues: • Vulnerability history shows that embedded systems are lacking good practices to a high degree.
Damage	Destruction of Hardware • Overheating • Explosion • "Bricking", e.g. destruction of firmware • Disabling of interfaces	• MinHWAccess • LockLogicalAccess	Assets (partially) not covered: • Firmware update interfaces Issues: • Lack of validation of input settings/firmware bundles. • Logical functions with potential for physical damage carried out in a logical/digital way, no possibility to implement physical safety switches.
	Waste/destruction of Resources • Excessive Heating/use of heat - producing resources • Excessive energy consumption • Excessive use of water/physical resources controlled by a computing control system	• SecDefaults • MinHWAccess • LockLogicalAccess	Assets (partially) not covered: • - Issues: • Vulnerability history shows that embedded systems are lacking good practices to a high degree.

Failures or malfunctions	Malfunction • Failure of medical devices • Overheating/explosion of batteries • Failure of control/production systems • Failure of access systems • Failure of alarm systems • Outage of monitoring system	• SDLC • SecStandards • OpenSecVal • PlatformSec • SecDefult	Assets (partially) not covered: • - Issues: • Vulnerability history shows that embedded systems are lacking good practices to a high degree.
	Modification-of-Service • Wrong treatment by medical devices • False negative reporting by alarm/monitoring systems • Granted access for unauthorized parties by access control systems	• SDLC • SecStandards • OpenSecVal • PlatformSec • SecDefult	Assets (partially) not covered: • - Issues: • Vulnerability history shows that embedded systems are lacking good practices to a high degree.
Outages	Denial-of-Service • Flooding/volumetric attack • Software bug/exploit • Logical flaw	• SDLC	Assets (partially) not covered: • - Issues: • Vulnerability history shows that embedded systems are lacking good practices to a high degree.
	Jamming	• -	Assets (partially) not covered: • - Issues:

			• -
Legal	Loss of Compliance • Voidance of certification/validation approvals • Violation of contractual requirements • Violation of internal/external compliance requirements • Violation of data protection laws	• SDLC • SUM • SecStandards • OpenSecVal	Assets (partially) not covered: • Hardware • Firmware Issues: • Certification processes lack flexibility to react to system updates in a timely manner with reasonable operational effort.

Table 6: Gaps in the protection of assets

The following structured list provides a more detailed explanation of the main identified gaps:

- **Platform security mechanisms:** Platforms for the operation of firmware/embedded platforms often lack security features or the computing resources to provide security features. This covers both exploit mitigation techniques such as DEP or ASLR as well as the possibility to lock access to certain memory regions or hardware functions. In addition, resources describing all security features (on any level) per platform are missing, resulting in a lack of use of the available mechanisms.
    - **IOMMU:** Many platforms comprise I/O memory management units which allow fine-grained access control for memory areas. However, few non-hypervisor systems make use of this capability to restrict access of devices/firmware to dedicated memory regions. (refer e.g. to [79])
- **Type-safe languages:** The use of type-safe languages/programming languages with embedded security features is feasible for embedded/firmware/hardware development, however, the use is not wide-spread. The availability of tool chains and development practices must be evaluated and potentially be improved.
- **Logical tamper detection:** There are various good practices and suggested approaches towards logical tamper detection available, e.g. via the use of anomaly detection, timing analysis, or physical co-processors. However, those good practices are not integrated into platform features, frameworks/libraries, or tools that can be integrated into firmware/hardware with reasonable effort. This results in a lack of protection by typical security solutions which are deployed on most systems.
- **Focus on Bios/CPU firmware:** Modern x86/x64 architectures (especially based on UEFI) provide various platform security features to both lock down a system, prevent tampering, and even detect tampering to a certain degree. However, the focus of those mechanisms is on the CPU and BIOS/UEFI. Other system firmware (e.g. for chipsets, graphic cards, or network interface cards) is not covered by those mechanisms.
- **Industry Standards:** There are various standardization bodies covering IT Security in general (such as ISO or NIST) and embedded systems/hardware development in general, however, security aspects of hardware-related assets is still a new field. The development of standards or establishment of industry/de-facto standards is just starting and needs to be progressed to be able to provide comprehensive baseline protection levels.

8. Recommendations

The gathered good practices and performed gap analysis shows that comprehensive guidance on hardware security is available. The following sub-sections provide specific recommendations on the use of and need for good practices as well as approaches how the identified gaps can be closed in the future.

Integration of threat analysis: System developers must integrate threat analysis aspects into every step of the development. The good practice *Secure Embedded Design and Development Lifecycle* must take hardware-specific aspects, such as trust boundaries within a single system, into account. These granular trust boundaries also result in the need to take stock of all available platform security and locking mechanisms to restrict access across the trust boundaries as much as possible.

- Stakeholders: System, platform, and firmware developers.
- Addressed Gaps:
    - Nefarious Activity/Abuse - Firmware Modification
    - Nefarious Activity/Abuse - Remote Firmware Attacks
    - Eavesdropping/ Interception/ Hijacking - Traffic Sniffing
    - Failures or malfunctions - Malfunction
    - Failures or malfunctions - Modification-of-Service

Language security aspects: During development, language security aspects should be taken into account. It should be evaluated whether languages such as Go or Rust can be used instead of C/C++ which are more prone to the introduction of memory corruption vulnerabilities.

- Stakeholders: System and firmware developers.
- Addressed Gaps:
    - Nefarious Activity/Abuse - Firmware Modification
    - Nefarious Activity/Abuse - Remote Firmware Attacks
    - Eavesdropping/ Interception/ Hijacking - Traffic Sniffing
    - Failures or malfunctions - Malfunction
    - Failures or malfunctions - Modification-of-Service

Use and contribute to standards: While there are efforts to provide industry guidance on the security of hardware-related assets, those efforts show certain gaps and need to be progressed to cover all aspects of secure development. Thus the existing standards should be used for the development and at the same time, gaps should be documented and closed by providing feedback and input for the standards. The contribution to the standards should be performed in accordance to all other good practices identified, in particular allowing for timely secure update management even though updates potentially change the system under standardization.

- Stakeholders: System, platform, and hardware/firmware developers.
- Addressed Gaps:
    - Legal - Loss of Compliance

Update management from the start on: The good practice *Secure Update Management* and the corresponding sources show the challenges for a comprehensive and secure update process for embedded devices. These challenges must be taken into account through the complete development process of the system as various design decisions (such as the use of some kind of HSM or the sizing of computing resources) is difficult to change in later stages. The sample execution of the update mechanisms should be a mandatory test for each development stage.

- Stakeholders: System, platform, and firmware developers.
- Addressed Gaps:
    - Nefarious Activity/Abuse - Firmware Modification
    - Nefarious Activity/Abuse - Remote Firmware Attacks
    - Eavesdropping/ Interception/ Hijacking - Traffic Sniffing
    - Failures or malfunctions - Malfunction
    - Failures or malfunctions - Modification-of-Service

Design for transparency and validation: While the design of hardware is often very sensitive intellectual property, the embedded security mechanisms should not be. Customers should be enabled to verify the effectivity of implemented security controls and to enable this, certain requirements for transparency, documentation, and verification tools must be integrated from the beginning of the development process on.

- Stakeholders: System, platform, and hardware/firmware developers.
- Addressed Gaps:
    - Nefarious Activity/Abuse - Firmware Modification
    - Nefarious Activity/Abuse - Remote Firmware Attacks
    - Physical Attacks - Property Losses
    - Failures or malfunctions - Malfunction

Design for tamper detection: One of most relevant described gaps is the intrinsically hard problem to detect modifications of hardware/hardware parts of a system/firmware when the tamper detection would depend on the functionality of this very hardware/system. Different approaches for tamper detection are described above; it must be evaluated early on which mechanisms are appropriate for the system to be developed (e.g. based on functionality characteristics or protection requirements).
Developers/manufacturers of hardware must spend relevant resources on the analysis whether and how tamper detection (not on the silicone level but with a focus as described above, e.g. detecting unauthorized firmware tampering) features can be provided by the system to be developed.

- Stakeholders: System, platform, and firmware developers.
- Addressed Gaps:
    - Nefarious Activity/Abuse - Attack Persistence
    - Physical Attacks - Hardware Modification

Integrate security aspects into selection criteria: As relevant factors in the regulation of markets, users and customers must demand security features and posture as well as a transparent validation of those from vendors. Security features and posture of hardware-related assets must be incorporated into selection criteria. In addition, end users should inform themselves about available security features and available updates on platforms where such a mind-set was not required before.

- Stakeholders: Users and customers.
- Addressed Gaps:
    - Damage - Destruction of Hardware
    - Damage - Waste/destruction of Resources
    - Outages - Denial-of-Service
    - Physical Attacks - Property Losses

9. Conclusions

The hardware-related threats and assets discussed in this threat landscape have been present for many years and are not resulting from the introduction of new technology but rather the focus on new ways of attacks. While this can be a result of the fast spread of computing devices through every aspect of life (often summarized as the Internet of Things) and the heavy daily use (even reliance on) personal computing devices, the performed threat, good practice, and gap analysis shows that reasonable and reliable security good practices are available, however not widely implemented yet.

In this report, we have attempted to create awareness for the changing threat landscape introduced by the growing amount of computing aspects throughout daily life and operations and, even more important, attempted to gather available sources of good practice and security guidance for the design, development, and use of mobile/personal/embedded computing devices.

Annex A: Sources

ID	NAME OF THE DOCUMENT/SOURCE	INFORMATION ITEM (E.G. URL OR FILE)	YEAR/VERSION	KEYWORDS
[1]	Inside the NSA's Leaked Catalog of Surveillance Magic	http://arstechnica.com/information-technology/2013/12/inside-the-nsas-leaked-catalog-of-surveillance-magic/	2013	Hardware Implant, Surveillance
[2]	F-Secure Labs: *Sharking: High-Rollers in the Crosshairs*	https://www.f-secure.com/weblog/archives/00002647.html	10.12.2013	Hardware Implant, Surveillance
[3]	NSA Ant Catalog Description	https://en.wikipedia.org/wiki/NSA_ANT_catalog		Hardware Implant, Surveillance
[4]	NSA Playset	http://www.nsaplayset.org/		Hardware Implant, Surveillance
[5]	Building a Reliable SMM Backdoor for UEFI	http://blog.cr4.sh/2015/07/building-reliable-smm-backdoor-for-uefi.html	2015	Firmware Implant
[6]	Samsung Recall Galaxy Note 7	http://www.samsung.com/us/note7recall/?CID=AFL-hq-mul-0813-11000170	2016	Physical Impact
[7]	Australian Government, Department of Defence, Defence Science and Technology Organisation – Hardware Trojans – Prevention, Detection, Countermeasures	http://www.dtic.mil/get-tr-doc/pdf?AD=ADA547668	2011	Hardware/Firmware Implant, Supply Chain
[8]	Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, Dennis Sylvester – Analog Malicious Hardware	http://ieee-security.org/TC/SP2016/papers/0824a018.pdf	2016	Hardware Implant
[9]	Samuel T. King, Joseph Tucek, Anthony Cozzie, Chris Grier, Weihang Jiang, and Yuanyuan Zhou – Designing and implementing malicious hardware	http://www.acoz.net/pubs/malicious_processor.pdf	2008	Hardware Implant
[10]	Cynthia Sturton, Matthew Hicks, David Wagner, Samuel T. King – Defeating UCI: Building Stealthy and Malicious Hardware	https://spqr.eecs.umich.edu/courses/cs660sp11/papers/defeating-uci-oak11.pdf	2011	Hardware Implant
[11]	Hardware attacks, backdoors and electronic component qualification	http://resources.infosecinstitute.com/hardware-attacks-backdoors-and-electronic-component-qualification/		Hardware Implant
[12]	Christian Krieg, Edgar Weippl – Malware in Hardware Infrastructure Components	https://www.sba-research.org/wp-content/uploads/publications/weippl_chapter.pdf		Firmware Implant, Hardware Implant
[13]	Adam Waksman, Simha Sethumadhavan – Silencing Hardware Backdoors	http://www.cs.columbia.edu/~simha/preprint_oakland11.pdf	2011	Firmware Implant, Hardware Implant

[14]	Peter Laackmann, Marcus Jahnke – Hardware-Trojaner als unterschätzte Gefahr	http://www.heise.de/newsticker/meldung/32C3-Hardware-Trojaner-als-unterschaetzte-Gefahr-3056452.html	2015	Hardware Implant
[15]	Xiaoxiao Wang, Mohammad Tehranipoor, Jim Plusquellic – Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions	http://www.engr.uconn.edu/~tehrani/publications/host08-1.pdf	2008	Hardware Implant
[16]	Stopping Hardware Trojans in Their Tracks	http://spectrum.ieee.org/semiconductors/design/stopping-hardware-trojans-in-their-tracks	2015	Hardware Implant
[17]	Ted Huffmire – Designing Secure Systems on Reconfigurable Hardware	http://cseweb.ucsd.edu/~kastner/papers/todaes08-design_secure.pdf	2008	Hardware Security
[18]	Secure Hardware Design for Trust	http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA597771	2014	Hardware Security
[19]	Georg T. Becker, Francesco Regazzoni, Christof Paar, Wayne P. Burleson – Stealthy Dopant-Level Hardware Trojans		2013	Hardware Security, Hardware Implant
[20]	Raghavan Kumar, Philipp Jovanovice, Wayne Burleson, Ilia Poliane – Parametric Trojans for Fault-Injection Attacks on Cryptographic Hardware	https://eprint.iacr.org/2014/783.pdf	2014	Hardware Implant
[21]	Practical Secure Hardware Design for Embedded Systems	http://www.grandideastudio.com/wp-content/uploads/secure_embed_paper.pdf		Hardware Security
[22]	Camel Tanougast, Abbas Dandache, Mohamed Salah Azzaz and Said Sadoudi (2012). Hardware Design of Embedded Systems for Security Applications	http://cdn.intechopen.com/pdfs-wm/31923.pdf	2012	Hardware Security
[23]	Mark Karpovsky – Design of Secure and Reliable Hardware	http://mark.bu.edu/EC500/lec_sprg2013/Lecture%201.pdf	2013	Hardware Security
[24]	Joe Grand – The Pitfalls and Perils of Poor Security	http://grandideastudio.com/wp-content/uploads/pitfalls_of_poor_security_slides.pdf		Hardware Security
[25]	Marko Wolf, Timo Gendrullis – Design, Implementation, and Evaluation of a Vehicular Hardware Security Module	http://evita-project.org/Publications/WG11.pdf		Hardware Security
[26]	OWASP Top IoT Vulnerabilities	https://www.owasp.org/index.php/Top_IoT_Vulnerabilities		Hardware/Firmware Security
[27]	Cloud Security Alliance – Future-proofing the Connected World: 13 Steps to Developing Secure IoT Products	https://downloads.cloudsecurityalliance.org/assets/research/internet-of-things/future-proofing-the-connected-world.pdf		Hardware/Firmware Security

[28]	PCILeech: DMA over PCI Express	https://github.com/ufrisk/pcileech		Hardware Attack
[29]	DMA attacking over USB-C and Thunderbolt 3	http://blog.frizk.net/2016/10/dma-attacking-over-usb-c-and.html	2016	Hardware Attack
[30]	Qin Long, Zhan Gao – Unified Extensible Firmware Interface (UEFI): Best Platform Security Practices	https://firmware.intel.com/sites/default/files/uefi-best-platform-security-practices%5B1%5D.pdf		Firmware Security
[31]	Intel Advanced Threat Research Team – Security below the OS with CHIPSEC Framework	http://www.intelsecurity.com/advanced-threat-research/content/chipsec.html	2016	Firmware Security
[32]	Intel Advanced Threat Research Team – Attacking and Defending BIOS in 2015	http://www.intelsecurity.com/advanced-threat-research/content/AttackingAndDefendingBIOS-RECon2015.pdf	2015	Firmware Security
[33]	Intel Advanced Threat Research Team – Attacking Hypervisors via Firmware and Hardware	http://www.intelsecurity.com/advanced-threat-research/content/AttackingHypervisorsViaFirmware_bhusa15_dc23.pdf	2015	Firmware Security
[34]	Intel Advanced Threat Research Team – A Tour Beyond BIOS Launching a VMM in EFI Developer Kit II	https://firmware.intel.com/sites/default/files/A_Tour_Beyond_BIOS_Launching_VMM_in_EFI_Developer_Kit_II_0.pdf		Firmware Security
[35]	Xeno Kovah, Corey Kallenberg – Are You Giving Firmware Attackers a Free Pass?	https://www.rsaconference.com/writable/presentations/file_upload/hta-f02-are-you-giving-firmware-attackers-a-free-pass_final.pdf	2015	Firmware Security
[36]	Corey Kallenberg, Xeno Kovah, John Butterworth, Sam Cornwell – ALL YOUR BOOT ARE BELONG TO US	https://cansecwest.com/slides/2014/AllYourBoot_csw14-mitre-final.pdf	2014	Firmware Security
[37]	Embedded Systems Conference Silicon Valley: Loren K. Shade – Implementing Secure Remote Firmware Updates	https://www.allegrosoft.com/wp-content/uploads/Secure-Firmware-Updates-Paper.pdf	2011	Firmware Security
[38]	Tony Mangefeste – Secure Firmware Considerations	http://www.uefi.org/sites/default/files/resources/UEFI_Secure_Firmware_Considerations_v2.pdf		Firmware Security
[39]	Dick Wilkins – UEFI Firmware Security Best Practices	http://www.uefi.org/sites/default/files/resources/2014_UEFI_Plugfest_06_Phoenix.pdf	2014	Firmware Security
[40]	OWASP – Embedded Application Security	https://www.owasp.org/index.php/OWASP_Embedded_Application_Security		Firmware Security
[41]	ENISA Threat Landscape 2015	https://www.enisa.europa.eu/publications/etl2015	2015	

[42]	Loïc Duflot, Yves-Alexis Perez, Guillaume Valadon, Olivier Levillain – Can you still trust your network card?	http://www.ssi.gouv.fr/uploads/IMG/pdf/csw-trustnetworkcard.pdf	2010	Firmware Exploit
[43]	Intel® Software Guard Extensions (Intel® SGX)	https://software.intel.com/en-us/sgx		Platform Security
[44]	Charlie Miller – Battery Firmware Hacking	https://www.defcon.org/images/defcon-19/dc-19-presentations/Miller/DEFCON-19-Miller-Battery-Firmware-Hacking.pdf	2011	Firmware Security, Firmware Exploit
[45]	Intel Active Management Technology (AMT)	http://www.intel.com/content/www/us/en/architecture-and-technology/intel-active-management-technology.html		Embedded Management Functionality
[46]	Joe Grand – Practical Secure Hardware Design for Embedded Systems	http://www.grandideastudio.com/wp-content/uploads/secure_embed_paper.pdf	2004	Firmware Security
[47]	Bruce Schneier – Security Risks of Embedded Devices	https://www.schneier.com/blog/archives/2014/01/security_risks_9.html	2014	Hardware Risks
[48]	Philip Koopman – Avoiding the Top 43 Embedded Software Risks	https://users.ece.cmu.edu/~koopman/pubs/koopman11_escsv_handouts.pdf	2011	Firmware Security
[49]	The Secret Behind the NSA Breach: Network Infrastructure Is the Next Target	http://www.darkreading.com/perimeter/the-secret-behind-the-nsa-breach-network-infrastructure-is-the-next-target/a/d-id/1326729	2016	Embedded Firmware Attacks
[50]	Yan Michalevsky, Dan Boneh – Gyrophone: Recognizing Speech from Gyroscope Signals	https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-michalevsky.pdf	2014	Embedded Exploitation
[51]	Kim Zetter – Why Firmware Is So Vulnerable to Hacking, and What Can Be Done About It	https://www.wired.com/2015/02/firmware-vulnerable-hacking-can-done/	2015	Firmware Security
[52]	Ralf-Philipp Weinmann – Baseband Attacks: Remote Exploitation of Memory Corruptions in Cellular Protocol Stacks	Usenix Woot 2012, https://www.usenix.org/system/files/conference/woot12/woot12-final24.pdf	2012	Firmware Security, Remote Firmware Attacks
[53]	Semtek Security Policy for Hardware and Firmware Designs Cipher Cryptographic Module (Version 1.00)	http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp953.pdf	2008	Embedded Security
[54]	Fred B. Schneider, Dexter Kozen, Greg Morrisett, Andrew C. Myers – Language-Based Security for Malicious Mobile Code	https://www.cs.cornell.edu/fbs/publications/langBasedMuriSmry.pdf	2005	Language Security
[55]	Ronald G. Minnich, Google; Andrey Mirtchovski, Cisco – U-root: A Go-based, Firmware Embeddable Root File System with On-demand Compilation	Usenix 2015: https://www.usenix.org/system/files/conference/atc15/atc15-paper-minnich.pdf	2015	Language Security

[56]	Kaspersky – Equation: The Death Star of Malware Galaxy	https://securelist.com/blog/research/68750/equation-the-death-star-of-malware-galaxy/	2015	Embedded Platform Insecurity
[57]	Jeff Forristal – Hardware Involved Software Attacks	http://forristal.com/material/Forristal_Hardware_Involved_Software_Attacks.pdf		Hardware-related Attacks
[58]	NIST SP 800-147 – BIOS Protection Guidelines	http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-147.pdf	2011	Firmware Security
[59]	CanSecWest, Yuriy Bulygin – Evil Maid Just Got Angrier – Why Full-Disk Encryption With TPM is Insecure on Many Systems	https://cansecwest.com/slides/2013/Evil%20Maid%20Just%20Got%20Angrier.pdf	2013	Firmware Security
[60]	Rafal Wojtczuk, Alexander Tereshkin – Attacking Intel® BIOS	https://www.blackhat.com/presentations/bh-usa-09/WOJTCZUK/BHUSA09-Wojtczuk-AtkIntelBios-SLIDES.pdf	2009	Firmware Security
[61]	NIST SP 800-155 – BIOS Integrity Measurement Guidelines (Draft)	http://csrc.nist.gov/publications/drafts/800-155/draft-SP800-155_Dec2011.pdf	2011	Firmware Security
[62]	ATM Industry Association – ATM Software Security Best Practices Guide Version 3	https://www.atmia.com/files/Best%20Practices/ATMIA%20Best%20Practices%20v3.pdf	2014	Firmware Security
[63]	Yoni Allon – The Secret Behind the NSA Breach: Network Infrastructure Is the Next Target	http://www.darkreading.com/perimeter/the-secret-behind-the-nsa-breach-network-infrastructure-is-the-next-target/a/d-id/1326729	2016	Hardware-related Attacks
[64]	Thomas Wollinger, Jorge Guajardo, Christof Paar – Cryptography in Embedded Systems: An Overview	https://www.emsec.rub.de/media/crypto/veroeffentlichungen/2011/01/21/wollingeretalembeddedworld2003.pdf	2003	Embedded Cryptography
[65]	BSI – Überblickspapier Smartphone	https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Download/Ueberblickspapier_Smartphone_pdf.pdf?__blob=publicationFile		Mobile Device Security
[66]	Gautam Khattak, Philip Koopman – Embedded System Code Review Checklist	https://users.ece.cmu.edu/~koopman/pubs/code_review_checklist_v1_00.pdf	2012	Embedded Security
[67]	Barr Group – a Embedded C Coding Standard		2013	Embedded Security
[68]	HD Moore – CVE-2015-7755: Juniper ScreenOS Authentication Backdoor	https://community.rapid7.com/community/infosec/blog/2015/12/20/cve-2015-7755-juniper-screenos-authentication-backdoor	2016	Embedded Backdoors

[69]	Lenovo – System Management Mode (SMM) BIOS Vulnerability	https://support.lenovo.com/de/en/solutions/LEN-8324	2016	Embedded Backdoors
[70]	Dinakar Dhurjati, Sumant Kowshik, Vikram Adve, Chris Lattner – Memory Safety Without Garbage Collection for Embedded Applications	https://pdfs.semanticscholar.org/e388/b64fa308d83cac6ba03baae3840e378a792e.pdf	2005	Language Security
[71]	Kerry Maletsky – RSA vs ECC Comparison for Embedded Systems	http://www.atmel.com/Images/Atmel-8951-CryptoAuth-RSA-ECC-Comparison-Embedded-Systems-WhitePaper.pdf	2015	Embedded Cryptography
[72]	Alexander Tereshkin, Rafal Wojtczuk – Introducing Ring -3 Rootkits	https://www.blackhat.com/presentations/bh-usa-09/TERESHKIN/BHUSA09-Tereshkin-Ring3Rootkit-SLIDES.pdf	2009	Firmware Security
[73]	ISACA – Firmware Security, Risks and Mitigation, Enterprise Practices and Challenges	http://www.isaca.org/Knowledge-Center/Research/Documents/CSX-Firmware_whp_eng_1016.pdf?regnum=341277	2016	Firmware Security
[74]	Core Collapse – A Real SMM Rootkit: Reversing and Hooking BIOS SMI Handlers	http://phrack.org/issues/66/11.html	2009	Firmware Security
[75]	Shawn Embleton, Sherri Sparks, Cliff Zou – SMM Rootkits: A New Breed of OS Independent Malware	http://www.co-c.net/repository-securite-informatique/Papers/SMM-Rootkits-Securecom08.pdf	2008	Firmware Security
[76]	Xeno Kovah, John Butterworth, Corey Kallenberg, Sam Cornwell – Copernicus 2: SENTER the Dragon!	https://www.mitre.org/sites/default/files/publications/Copernicus2-SENTER-the-Dragon-CSW-.pdf	2014	Firmware Security
[77]	Nick Petroni, Timothy Fraser, Jesus Molina, William Arbaugh – Copilot - a Coprocessor-based Kernel Runtime Integrity Monitor	Usenix 2004: https://www.usenix.org/legacy/event/sec04/tech/full_papers/petroni/petroni_html/main.html	2004	Tamper Detection
[78]	Bill Jacobs, Cisco Systems, Inc., Vincent J. Zimmer, Intel Corporation – Open platforms and the impact of Security Technologies, Initiatives, and Deployment Practices	https://firmware.intel.com/sites/default/files/resources/Platform_Security_Review_Intel_Cisco_White_Paper.pdf	2012	Platform Security
[79]	Jacob Torrey – The foundation is rotting and the basement is flooding: A deeper look at the implicit trust relationships in your organization	https://www.troopers.de/media/filer_public/06/b0/06b02356-df68-4c2a-98e8-e94f2f1078c4/troopers15_torrey.pdf	2015	Firmware Security
[80]	Steve Heath – Embedded Systems Design	ISBN: 978-0-7506-5546-0	2002	Embedded Devices

[81]	KrebsOnSecurity Hit With Record DDoS	https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/	2016	Embedded Botnet
[82]	DDoS botnets built using Linux malware for embedded devices	http://www.computerworld.com/article/3090153/security/ddos-botnets-built-using-linux-malware-for-embedded-devices.html	2016	Embedded Botnet
[83]	DDoS Attack Takes Down Central Heating System Amidst Winter In Finland	http://thehackernews.com/2016/11/heating-system-hacked.html	2016	Infrastructure Outage
[84]	PHLASHING DENIAL OF SERVICE ATTACK, THE NEW HYPE	http://hackaday.com/2008/05/20/phlashing-denial-of-service-attack-the-new-hype/	2008	Denial-of-Service
[85]	Jonas Zaddach, Anil Kurmus, Davide Balzarotti, Erik-Oliver Blass, Aurelien Francillon, Travis Goodspeed, Moitrayee Guptak, Ioannis Koltsidas – Implementation and Implications of a Stealth Hard-Drive Backdoor	https://www.ibr.cs.tu-bs.de/users/kurmus/papers/acsac13.pdf	2013	Firmware Modification
[86]	Charlie Miller, Chris Valasek – Remote Exploitation of an Unaltered Passenger Vehicle	http://illmatics.com/Remote%20Car%20Hacking.pdf	2015	Firmware Modification, Remote Firmware Attack
[87]	Analysis of an Alarm System	https://insinuator.net/2015/04/analysis-of-an-alarm-system/ https://insinuator.net/2015/05/analysis-of-an-alarm-system-part-23/	2015	Access Control Bypass
[88]	Experts Fear Possible Voting Machine Tampering in November	http://www.cnsnews.com/news/article/alex-grubbs/experts-fear-possible-voting-machine-tampering-november	2016	Tampering, Compliance
[89]	Water treatment plant hacked, chemical mix changed for tap supplies	http://www.theregister.co.uk/2016/03/24/water_utility_hacked/	2016	Control System Attack
[90]	Graeme Neilson, Enno Rey – Supply Chain (In-) Security	https://www.ernw.de/wp-content/uploads/Day-Con_Aura_ERNW_Supply_Chain_Insecurity.pdf	2010	Supply Chain Security

Annex B: Detailed Mind Map for Hardware-related Threats

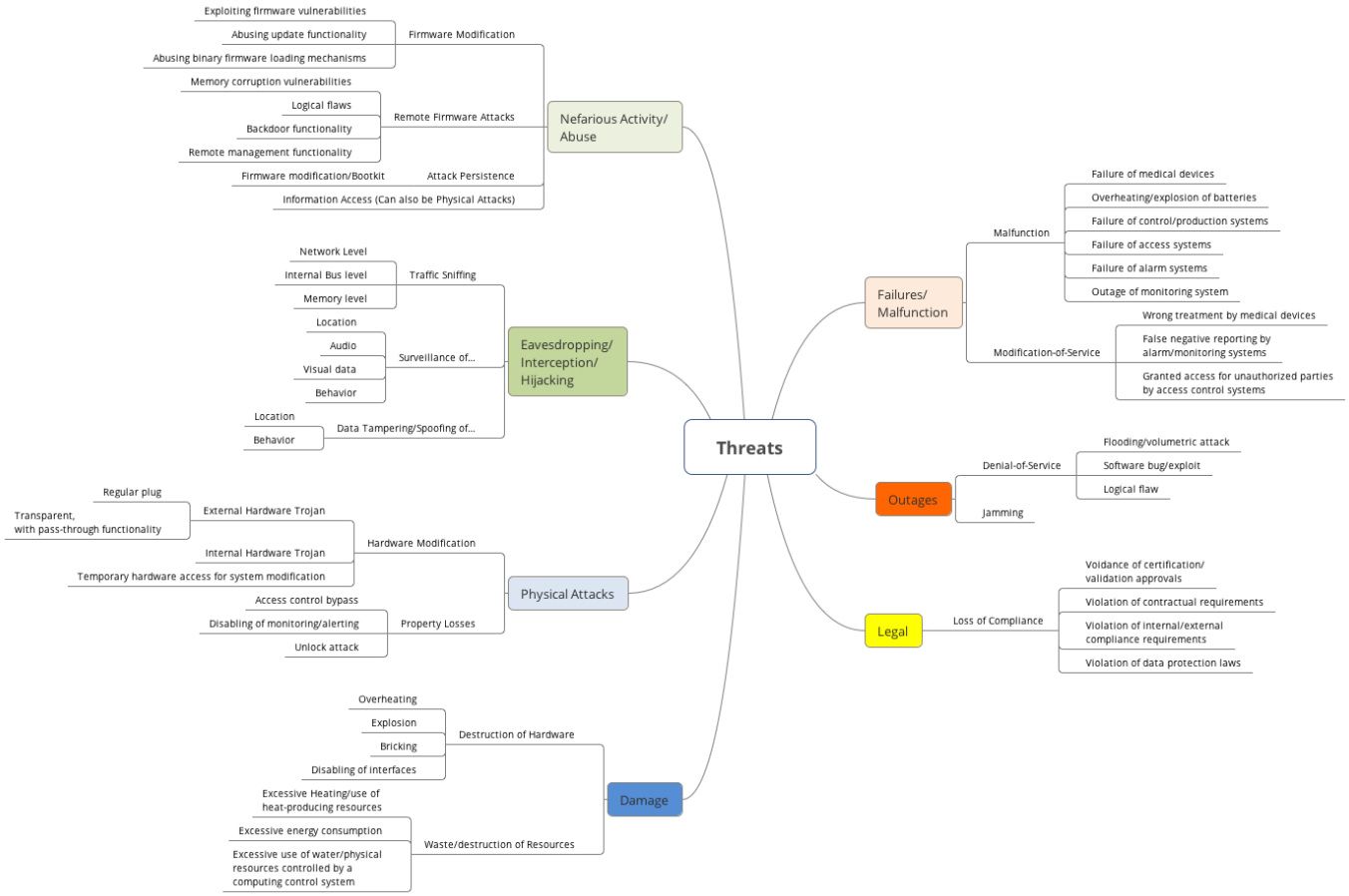

Figure 4: The assumed threats for Hardware

ENISA

European Union Agency for Network
and Information Security
Science and Technology Park of Crete (ITE)
Vassilika Vouton, 700 13, Heraklion, Greece

Athens Office

1 Vass. Sofias & Meg. Alexandrou
Marousi 151 24, Athens, Greece

PO Box 1309, 710 01 Heraklion, Greece
Tel: +30 28 14 40 9710
info@enisa.europa.eu
www.enisa.europa.eu