Hardware security, vulnerabilities, and attacks: a comprehensive taxonomy

Hardware Security, Vulnerabilities, and Attacks: A Comprehensive Taxonomy*

Paolo Prinetto and Gianluca Roascio

Cybersecurity National Laboratory, Consorzio Interuniversitario Nazionale per l'Informatica
Dipartimento di Automatica e Informatica, Politecnico di Torino, Turin, Italy
paolo.prinetto@polito.it
gianluca.roascio@polito.it

Abstract

Information Systems, increasingly present in a world that goes towards complete digitalisation, can be seen as complex systems at the base of which is the hardware. When dealing with the security of these systems to stop possible intrusions and malicious uses, the analysis must necessarily include the possible vulnerabilities that can be found at the hardware level, since their exploitation can make all defences implemented at web or software level ineffective. In this paper, we propose a meaningful and comprehensive taxonomy for the vulnerabilities affecting the hardware and the attacks that exploit them to compromise the system, also giving a definition of Hardware Security, in order to clarify a concept often confused with other domains, even in the literature.

1 Introduction

Every process and even every object that surrounds us is today managed by computing systems, and every day an endless amount of data is produced by these devices and exchanged with the external world, thanks to advanced connection capabilities. Yet, right because of this deep digitisation of our lives, this information sea is full of sensitive data that reveal our most private aspects. Innovation, at least initially, has not taken into account security and privacy issues, which instead must be seriously addressed.

According to US National Institute of Standards and Technology (NIST)[1], information security is defined as "*the protection of information and information systems from unauthorised access, use, disclosure, disruption, modification, or destruction*" [3], i.e., the protection against any misuse of Information Systems *assets*, which can be information itself or properties of the system.

Information security is declined in three different basic concepts: *Confidentiality, Integrity* and *Availability*, usually referred to as the *CIA triad*. In particular, *Confidentiality* refers to granting the access to assets only to those who are authorised, *Integrity* refers to maintaining assets unchanged between authorised accesses to them, and *Availability* refers to ensuring the access to assets when requested.

Information Systems can be viewed as complex systems consisting of multiple *layers*, as shown in Figure 1. Each layer treats information relying on *facilities* provided by the underlying layer. At the top of all layers is the *user* of the Information System. The highest layer is the *communication* layer, which through network and web services allows the distributed processing of information by *application software* running on different systems. In order to

[1] https://www.nist.gov

work, the applications need services provided by the *system software* (typically the Operating System), which in turn is the last virtualisation layer on top of the *hardware*. "Hardware" is a vague concept, being it often given different interpretations and meanings, as a consequence of the peculiar points of view of different stakeholders, including end-users, providers, OEMs, manufacturers, designers, etc. In the sequel, the term *hardware* is used to collectively refer to the whole set of electronic devices used to set-up an Information System, Information Technology (IT) or Operational Technology (OT) indifferently, regardless its complexity, its field of application, and the functionality/role of the devices within it.

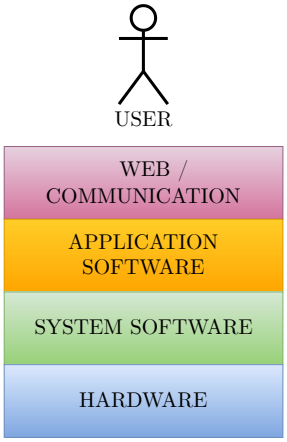

Figure 1: Layerized view of a computing system.

From the security point of view, any component of any Information System layer may have *weaknesses* that can generate *vulnerabilities*. The MITRE Corporation[2] defines a vulnerability as a weakness present inside a component of an information system that, "*when exploited, results in a negative impact to Confidentiality, Integrity, OR Availability*" [2]. Anything that endangers at least one of the three aspects of the CIA triad makes the system vulnerable, i.e., not completely secure.

When a component of one of the layers is compromised by an attack, either the lower layer provides protection, and thus the intrusion is stopped, or it is compromised as well, and the attacker can use it maliciously. It is therefore clear that the base of the layer stack, the hardware, plays a primary role in Information System security: it represents, by construction, *the last line of defense* against intrusions [7, Section 4.1]. Directly or indirectly being the base all the other layers rely on, if attacked, it may render *useless* all the defences implemented in the upper layers.

The presence of hardware vulnerabilities has thus an obvious impact on the Information System security, but this is not the only role that hardware plays in its security. We can, in fact, identify three different areas to consider, as shown in Figure 2: *Hardware Security*, *Hardware-based Security*, and *Hardware Trust*.

Hardware Security refers to all the actions needed to (i) identify hardware vulnerabilities, (ii) analyse their effects, (iii) prevent their exploitations by mitigating, reducing, and (ideally) making null the risks induced by their presence, (iv) develop and implement protections and remediation solutions, and (v) possibly avoid them by proper remediations during

[2] https://cve.mitre.org/

the design and production phases (*Security-by-Design*). Note that this definition is in no way constrained on *where* or *when* what described above can be done. For example, the fact that the vulnerabilities be located in the hardware and that the hardware attacks try to open breaches through them to compromise the security of the system, does not necessarily mean that the defences against them must be implemented at the hardware level. This would be extremely limiting, since most vulnerabilities are discovered once the hardware is already operating in the field, without the possibility of being patched, as it can mostly be done for software. Therefore, any technique aimed at counter hardware attacks falls under the definition of Hardware Security, even if mitigations are applied at the upper layers.

Hardware-based Security refers to all the solutions aimed at resorting to hardware to protect the system from attacks that exploit vulnerabilities present in other components of the system.

Hardware Trust refers to minimising the risks introduced by hardware counterfeiting, thus guaranteeing the other components of the system about the authenticity of the used hardware devices.

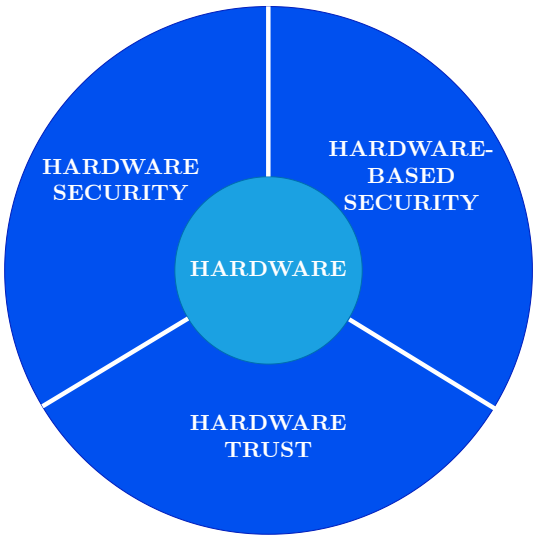

Figure 2: The role of Hardware in Information System security.

In the sequel of this paper, we shall zoom on Hardware Security, only, presenting a taxonomy of both vulnerabilities affecting the hardware and of the attacks targeting it. Section 2 contextualises the paper and shows some previous attempts to systematise the topic; then, Section 3 classifies hardware vulnerabilities, Section 4 presents hardware attacks and, eventually, Section 5 concludes the paper.

2 State of the Art

Since Information Systems began to spread and evolve, the topic of security has always been mainly addressed in relation to the protection from intrusions made possible by their web

connections, i.e., in an environment potentially open to anyone. It is therefore a fact that networks and software have received the most of the attention, while hardware has traditionally been considered as secure and inviolable. On the other hand, the role of hardware components in safety and in safety-critical applications have been deeply investigated [13]: it was commonly believed that hardware could at most *fail*, but not be attacked.

At the end of the last century, smart cards were already diffused. Based on chips specialized in security and authentication applications, these devices were considered impossible to crack if not with very advanced means, out of the possibility of common hackers. But starting from 1996, this thesis started to be dismantled through demonstrations, for the first time, of fault injection attacks or microprobing experiments carried out with common equipment against these chips [5] [6], and the problem began to be slowly acknowledged.

In the same years or a little later, important authors such as Kocher [33] [32] and others [41] [39] began to raise the problem of extrapolating information from secure devices such as smart cards simply by *listening* to the surrounding environment, e.g., by measuring the time taken, the energy consumed, the radiation emitted. Cryptographic algorithms, considered practically impossible to break mathematically, are instead vulnerable in their physical implementations. This was how the so-called *side-channel attacks* started to be known.

At the beginning of the century, the *vertical* integration model in the hardware supply chain was abandoned in favor of the *horizontal* one: instead of taking care of all stages of production, from specifications to final manufacture, companies started to outsource manufacturing to third-party companies, to which the layout of their devices is delivered. Therefore, the community started to reason about the possible risks of counterfeiting and piracy deriving from this, with a first article in 2001 by Koushanfar *et al.* [35]. The issue was even raised years later by the United States Congress [1]. Thus, a whole literature has been produced on the so-called *hardware metering* [34] and its implementation methods, including Physical Unconable Functions (PUFs) [40] or circuit obfuscation [42]. Similarly, a manufacturing process that includes untrusted actors started to raise doubts about the possibility of inclusion of *hardware Trojan horses* [55], i.e., Trojans inserted directly into the circuit, to be activated once the device is put into operation.

The concept of security related to hardware is therefore a young concept, and it may seem in itself a spurious union of techniques for protecting sometimes the originality and the integrity of the hardware design, sometimes the information itself treated by the hardware. Only in more recent years, some authors have tried to tidy up by proposing examples of taxonomies, among which we report here the most significant according to our opinion.

In a paper of 2014, Rostami et al. [45] distinguish, within the sphere of Hardware Security, 5 major issues: Hardware Trojans, Reverse Engineering of the design, Intellectual-Property Piracy, Side-Channel Attacks and Hardware Counterfeiting. It is a classification that confuses vulnerabilities, types of attacks and purposes of attacks, since, for example, many reverse-engineering attacks are certainly performed to steal the intellectual property of a circuit, while Trojans are to be considered rather as vulnerabilities, triggered later by an attack, but they are not properly an attack category.

In the same year, Hamdioui et al. [25] tried to classify attacks in *attacks to data* (e.g., Side-channel attacks), *attacks to design* (e.g., reverse-engineering attacks) and *attacks to functionality*, with three modes in the context of attacks to data: *invasive, non-invasive*, or *semi-invasive* with respect to the physical device itself, a very important concept that will be discussed later in the paper.

In their handbook published in 2018 [10], Bhunia and Tehranipoor well explain problems related to security of hardware components with many practical examples, without much refining the taxonomy of Rostami's 2014 article, but adding a fundamental distinction of the overall

problem in two wide families: (i) attacks *targeting hardware* with their countermeasures, and (ii) attacks *targeting the system* with their *hardware-based* countermeasures, i.e., what we have respectively called Hardware Security and Hardware-based Security in the previous Section.

3 Hardware Vulnerabilities Taxonomy

The proposed taxonomy of hardware vulnerabilities is shown in Figure 3. Vulnerabilities are first clustered according to their **nature** and their **domain**, in turns into different criteria.

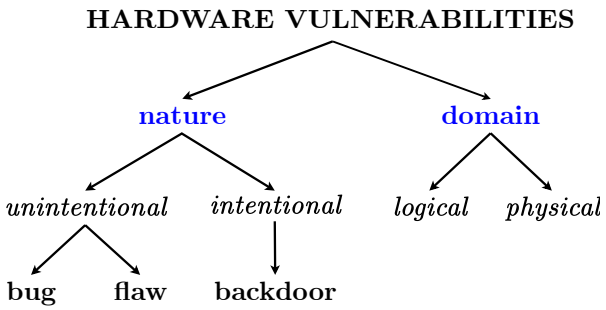

Figure 3: Hardware vulnerabilities taxonomy.

The nature may be ***intentional*** or ***unintentional***, i.e., the vulnerability may be introduced into the device voluntarily or not during its design and production phases. Unintentional vulnerabilities are further split into **bugs** and **flaws**.

A **bug** is an inconsistency between a specification and its actual implementation, introduced by a mistake during a specific design phase which is not detected during the subsequent V&V (Validation and Verification) phase.

A **flaw** is, instead, a non-primary feature that does not constitute an inconsistency w.r.t. the specs, and that is the result of a misconception of the designer who did not take into consideration its potential dangerousness. A flaw differs from a bug, being not colliding with any specification. As an example, in the design of modern microprocessors, the need to optimize performance through speculative execution and aggressive caching caused flaws such as the famous Meltdown [37] and Spectre [31]: such vulnerabilities were not born by a mistake made by the designer, but unintentionally introduced during the optimisation phase, without taking into account the risks that those race conditions could have led to.

A vulnerability inserted intentionally inside a hardware device can be referred to as a **backdoor**, as the person who inserts them wants to guarantee her/himself (or someone else) the possibility of a later access or misuse that is outside the set of intended use-cases. Note that the presence of a backdoor exposes the hardware component to threats independently of the fact it was inserted maliciously or not. From the one hand, an example of malicious backdoor is a *Hardware Trojan* [55], i.e., a rogue piece of circuitry inserted at a given point of the design and production phases, which can carry out unauthorised actions when its "triggering" conditions are satisfied. As already said, with the globalization of Integrated-Circuit (IC) design and manufacturing, the outsourcing of production task has become a common way to lower the product's cost. Embedded hardware devices are not always produced by the companies that design and sell them, nor in the same country where they will be used. A malicious intruder with access to the manufacturing process can introduce some changes to the final product. A

Hardware Trojan is characterized by a *payload*, i.e. the entire activity that the Trojan executes when it is activated, and by a *trigger* which is the condition verified in the state of the circuit that activates the payload. In general, malicious Trojans try to bypass or disable the security fence of a system, they can leak confidential information by radio emission or by other side-channel signal. A Trojan can also be used to disable, derange or destroy the entire chip or components of it. A Trojan can be introduced during any production step (design, fabrication, test, assembly) and at any level (register-transfer level, gate level, transistor level and even physical level).

From the other hand, an example of non-malicious backdoors is provided by the *undocumented instructions* of some processors belonging to x86 family, such as the one presented in [15]: the undocumented opcode `ALTINST` (`0x0F3F`), most likely originally introduced by the designers for debugging purposes, allows the user to switch to an alternative ISA (Instruction Set Architecture), closer to the actual inner RISC architecture, and it can be used maliciously to mount a privilege escalation attack.

Orthogonally to its nature, a hardware vulnerability belongs to a **domain**, either *logical* or *physical*. A hardware vulnerability is logical when it has been introduced during the early design phases of the device, whereas it is physical when it is related to vulnerabilities introduced during the latest technology-mapping steps of the design process.

A typical example is here provided by the fact that a series of consecutive write operations into a DRAM memory cell (*row hammering*) can induce adjacent cells to flip their content, due to an electric leakage effects [30]. Such a vulnerability is in fact intrinsic to the technology adopted for implementing the memory, even if an accurate analysis of the well known linked dynamic faults in DRAM [4] [14] could suggest proper remediation at the design time.

4 Hardware Attacks Taxonomy

For the very meaning of the term, a vulnerability is not such if it cannot be *exploited*, because it would not expose the system to any risk, so it would not constitute any weakness. The exploit is the mean or method of taking advantage of a vulnerability for malicious purposes. Therefore, a hardware attack can be defined as the act of taking advantage of a hardware vulnerability.

It is important to clearly point out that an attack always happens just when the hardware affected by a vulnerability is operating in the field: modifying a design to introduce a backdoor is a vulnerability insertion, while exploiting it is an attack.

Moreover, if the presence of a vulnerability jeopardises Confidentiality, Integrity or Availability (Section 3), and if the vulnerability is such only if it is exploitable, then an attack, using an exploit, is by definition an action that puts at risk the Confidentiality, the Integrity or the Availability of an asset, and therefore everything that does not impact on any of these three properties is outside the definition of attack.

The taxonomy for hardware attacks is summarised in Figure 4.

A hardware attack is first classified by the **goal** for which it is launched. The goal is the malicious action that the attacker wants to take against an asset of the attacked hardware, defined as a **target**. The target can be the ***information*** that the hardware is treating, but also a ***property*** of the hardware itself, either functional or non-functional [26]. One can launch an attack to:

steal a target (e.g., a cryptographic key, a secret password, an intellectual property, a resource, etc.); referring to the CIA triad, ***stealing*** is an action carried out in violation of Confidentiality, since the attacker takes possession of an asset of which she/he does not own the

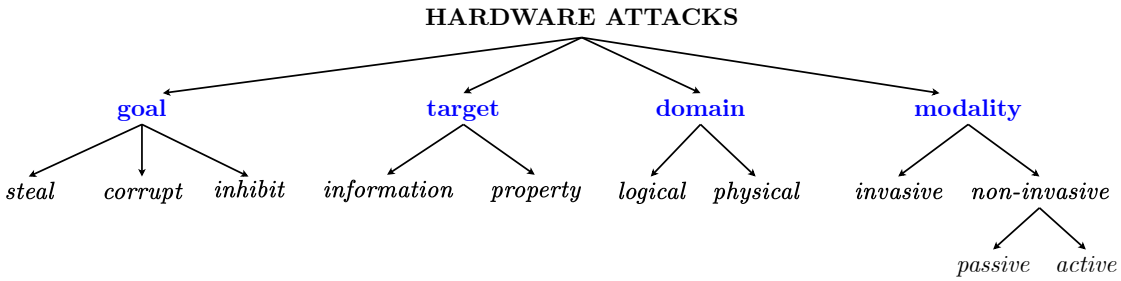

Figure 4: Hardware attacks taxonomy.

rights of access or use. It worths pointing out that the so called *intellectual property (IP) theft* is to be considered as a case of *IP-piracy attack*, and related solutions are demanded to Hardware Security. Intellectual property is in fact a full-fledged target according to the definition given in Section 4, and therefore it should be protected exactly as any other hardware asset;

corrupt a target (e.g., a memory word, a permission file, a functionality to make it folded to one's advantage, etc.); ***corrupting*** is an action carried out in violation of Integrity, since the attacker modifies an asset without being authorised to do it;

inhibit a target (e.g., a service, a set of critical data, a defense mechanism, etc.); ***inhibiting*** is an action carried out in violation of Availability, since the attacker prevents an asset from being properly accessed or used by those who hold rights to do that.

As well as vulnerabilities, hardware attacks always have a **domain** in which they are implemented. An attack belongs to the ***logical*** domain if it is implemented starting from upper layers with respect to hardware (Figure 1), i.e., when a hardware vulnerability, logical or physical, is exploited through actions not directly on the hardware itself, but on the software levels running on top of it. This domain includes, for example, *privilege escalation attacks* exploiting the *row-hammer* vulnerability [49] [56], or those that exploit vulnerabilities in processor microarchitecture such as Meltdown [37], Spectre [31] or others [20] [12] [47] [28], and also *cache-based attacks* [58] [48].

An attack belongs instead to the ***physical*** domain if it is implemented through actions directly performed on the attacked hardware device.

Finally, a hardware attack is qualified depending on the **modality** in which it is carried out. The attack is ***invasive*** when the actions taken against the attacked hardware includes physical intrusions such as desoldering, depackaging, disconnection of its internal components. Attacks having this modality are, for example:

- **Microprobing Attacks**: A microprobing attack tries to extract information by measuring electrical quantities directly on the silicon die of the target device, once obtained physical access to it. The die exposition is usually achieved by removing the plastic packages via chemical etching and/or by mechanical approaches. When possible, attackers study the netlist of the target before the attack, so with little reverse engineering they are able to find matches with the layout in order to locate connection carrying sensible

data. At this point, thank to advanced equipment as *Focused Ion Beam* (FIB) generators, they can obstruct wires with nanometric precision, or create conductive paths that serve as electrical probe contact in a further moment. A probe equipment is then employed to read the target signals and extract information. Such sophisticated equipment seems difficult to obtain commonly, but for example a FIB generator can be rented for just a couple hundred dollars per hour, which is reasonable with respect to an information theft that could be highly rewarding [50] [52].

- **Reverse Engineering Attacks**: An attack of reverse engineering is similar to micro-probing with respect to mounting phase (desoldering and decapsulation), but actually has a different scope. It in fact aims at understanding the structure of a semiconductor device and its functions, i.e., at stealing the intellectual properties of the designer. A deep knowledge and expertise on advance IC design are obviously required to succeed. All the layers formed during chip fabrication are removed one-by-one in reverse order and photographed to determine the internal structure of the chip. At the end, by processing all the acquired information, a standard netlist file can be created and used to simulate and eventually redesign the target device [19].

- **Data Remanence Attacks**: Computers typically store secret data in DRAM, properly de-powered when the device is tampered with. It is common to think that once the power is down, the content of volatile memories is erased (this is why they are called volatile, actually). Although, it has been proved that the charge stored in a DRAM cell has a given *decay rate* which is not infinitive and strictly depends on temperature. At temperatures from $-50°$ C down, the contents of RAMs can be "frozen" and kept for one or even more days. This is what usually happens in a *cold-boot attack* [23] [22], in which the hacker uses spray cans or liquid nitrogen on a volatile device just disconnected from the original system and gains precious time to perform a *memory dump*, i.e., a copy of the contents on a non-volatile device for subsequent analysis. Data remanence affects in a different way non-volatile types of memory such as EEPROM and Flash. Some sensible information thought to be erased can still be extracted [51].

The attack is instead ***non-invasive*** when it can be carried out without any physical contact with the device under attack. Non-invasive attacks are further split into *passive* and *active*. Passive non-invasive attacks are carried out by analysing and measuring one (or more) physical dynamic entities of the attacked hardware. All different types of *side-channel attacks* [36] [54] belong to this category. Active non-invasive attacks require instead specific actions on the device, aimed at forcing the system into abnormal states in which the goal is easier to reach. This category includes all the different types of *fault attacks* [11] [8] and *test-infrastructure-based attacks* [57] [44].

Side-Channel Attacks. Being something with physical consistence, when it is in activity, the hardware unintentionally releases in the surrounding environment a certain number of "clues", such as spent time, spent energy, electromagnetic radiation released, noise, etc.. These clues, along with the knowledge of some details about the device structure or just about the executed algorithms, may turn out to be critical for information protection. The mostly known classes of side-channel attacks are:

- *Timing Attacks*: A timing side-channel attack tries to recover sensible data by measuring their computation time in a piece of hardware. In most cases, the algorithm implementation strongly depends on the actual values of its input. If an attacker knows this

correlation, he can extract, for example, the encryption key or the password that is being processed. Examples of timing attacks against hardware implementations of RSA [33] or AES [9] [29] have been presented in literature.

- *Power Attacks*: The actual power consumption of a programmable device depends on both the executed instructions and the processed data. A power side-channel attack tries to read in reverse this process and to recover sensible data processed by measuring the variation of power consumption of the hardware device [32] [53].

- *Electromagnetic Attacks*: Whenever a current flows, an electromagnetic field is created around it. This radiation unintentionally carries information about the source, and by resorting to proper capturing devices, such as an induction coil, located in the proximity of the device, one can reconstruct the digital signal which originated it [59].

- *Acoustic Attacks*: Acoustic cryptoanalysis exploits vibration produced by hardware components of every kind and at any level, from device to circuit level. Covert listening devices may be placed by attackers to record the sound emitted by keyboards and keypads, and then a significative amount of sensed data can be later processed by signal analysis and/or Machine Learning algorithms to associate a particular sound-wave with the pressed key [43] [24]. Acoustic emissions in the ultrasonic band occur in circuit elements as coils and capacitors as a consequence of the current flowing through them. Voltage regulation circuits in PC motherboards are responsible for acoustic emanation which are directly correlated with CPU activity [21].

- *Optical Attacks*: Besides draining current or emitting radiation, a transistor that switches also emits some light in the form of a few photons for a very short time. If an attacker is able to detect such an emission, he can steel sensible information from the circuit [17]. Alternatively, information-carrying light emissions can also be exploited when LEDs are employed as device activity indicators [38].

Fault Attacks. They consist in the injection of deliberate (malicious) faults into the target device, aimed at bringing it into a set of states from which private internal information items can be fraudolently extracted. Types of fault attacks are mostly clustered according to the fault injection techniques. The most relevant are:

- *Supply Attacks*: If an attacker is able to tap into the power supply line of the target device and connect his power unit, he can underpower the device itself. If the power is lower, the delay of logic gates increases and in the case of critical paths it may happen that wrong values are sampled; this practically implies that one, or more, faulty bits, are injected into the system [16]. On the other hand, if a chip is overpowered, serious damaging actions can be carried out.

- *Clock Attacks*: The length of a single cycle can be shortened through forcing a premature toggling of the clock signal. In this way, registered bytes can be corrupted. To alter the length of the clock cycle, the attacker needs to get a direct control of the clock line, as it typically happens when smart cards are targeted. As an unplanned clock edge introduces a *glitch* in the internal signals, these attacks are also knowns as *Glitch Attacks* [18].

- *Heating Attacks*: Rising the temperature in the environment in which the target device operates may be exployted to attack it. Electrons inside the transistors are excited by the surrounding heat and random currents are generated, which may lead to bit flipping

(both in SRAM memory cells inside processors and in DRAM memory cells) or even to accelerated the ageing of the circuit, with the extreme consequence of its destruction when the overheating reaches a given threshold [27].

- *Radiation Attacks*: A practical way to induce faults without having to tap into the device is to cause strong electromagnetic disturbances near it. The eddy currents induced in the circuit by strong EM pulses cause temporary alterations of the level of a signal, which may be, for example, recorded by a latch or a flip-flop. When the disturbance becomes higher and higher, components of the device may stop working or even be physically destroyed [46].

Test-Infrastructure-Based Attacks. Hardware designers systematically rely on *Design-for-Testability* and *Built-in Self Test* (BIST) methodologies [13] to improve testability of the target system both at the end-of-production and in-field. Some of these methodologies are so widely adopted to become standards, both de-facto and de-iure. Examples include, among the many, IEEE 1149.1 (aka Boundary Scan) and IEEE 1500. Unfortunately, these test infrastructures, mandatory for getting the desired levels of testability in terms of cost, in most cases create severe security hazards. As an example, when the pins of an 1149.1 standard interface are left outside accessible, a potential attacker can easily exploit the scan chain to get the data stored into the connected flip-flops. Once the position of the target elements (e.g., registers containing secret keys) inside the chain are known, the attack is very easily accomplished [57] [44].

5 Conclusions

In this paper, we emphasized the importance of information security aspects related to hardware, and we have tried to characterize the roles that it has in the security domain. In fact, not everything related to "hardware" and "security" can be collected into the Hardware Security field, but should instead be distinguished. First, the hardware can be seen as a component to be secured, since it may contain vulnerabilities like any other component: this is the actual domain of Hardware Security. The hardware can also be seen as a *mean* by which to implement the system's security (Hardware-Based Security). On the other hand, there are several issues related to the Hardware Trust, which has to do with the authenticity of hardware components and the contrast to counterfeiting.

We have then proposed a definition of hardware vulnerability and hardware attack, providing for each of these two concepts a meaningful and comprehensive taxonomy. We classified vulnerabilities depending on their domain (logical, physical) and on their nature (intentional, unintentional). We then classified attacks depending on their target (information, property), their goal to be reached on the target (steal, corrupt, inhibit), the way they are carried out (invasively, non-invasively) and the domain in which they are implemented (logical, physical).

References

[1] Senate Armed Services Committee Releases Report on Counterfeit Electronic Parts. https://www.armed-services.senate.gov/press-releases/senate-armed-services-committee-releases-report-on-counterfeit-electronic-parts, 2012. [Online; accessed 16-January-2020].

[2] CVE - Terminology. https://cve.mitre.org/about/terminology.html, 2019. [Online; accessed 26-November-2019].

[3] Information Security - Glossary — CSRC. https://csrc.nist.gov/glossary/term/information-security, 2019. [Online; accessed 25-November-2019].

[4] Z. Al-Ars and A. J. van de Goor. Approximating infinite dynamic behavior for dram cell defects. In *Proceedings 20th IEEE VLSI Test Symposium (VTS 2002)*, pages 401–406, April 2002.

[5] R. Anderson and M. Kuhn. Tamper resistance-a cautionary note. In *Proceedings of the second Usenix workshop on electronic commerce*, volume 2, pages 1–11, 1996.

[6] R. Anderson and M. Kuhn. Low cost attacks on tamper resistant devices. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols*, pages 125–136, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[7] R. Baldoni, R. De Nicola, and P. Prinetto. *Il Futuro della Cybersecurity in Italia: Ambiti Progettuali Strategici*. Consorzio Interuniversitario Nazionale per l'Informatica - CINI, 2018. ISBN: 9788894137330.

[8] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache. Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures. *Proceedings of the IEEE*, 100(11):3056–3076, Nov 2012.

[9] D. J. Bernstein. Cache-timing attacks on AES. 2005.

[10] S. Bhunia and M. Tehranipoor. *Hardware Security: A Hands-on Learning Approach*. Morgan Kaufmann, 2018.

[11] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In *Annual international cryptology conference*, pages 513–525. Springer, 1997.

[12] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx. Foreshadow: Extracting the keys to the intel sgx kingdom with transient out-of-order execution. In *27th USENIX Security Symposium USENIX Security 18)*, pages 991–1008, 2018.

[13] M. Bushnell and V. Agrawal. *Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits*, volume 17. Springer Science & Business Media, 2004.

[14] S. Di Carlo and P. Prinetto. Models in memory testing. In *Models in Hardware Testing*, pages 157–185. Springer, 2010.

[15] C. Domas. Hardware backdoors in x86 cpus, 2018.

[16] P. Dusart, G. Letourneux, and O. Vivolo. Differential fault analysis on AES. In *International Conference on Applied Cryptography and Network Security*, pages 293–306. Springer, 2003.

[17] J. Ferrigno and M. Hlavac. When aes blinks: introducing optical side channel. *IET Information Security*, 2(3):94–98, Sep. 2008.

[18] T. Fukunaga and J. Takahashi. Practical fault attack on a cryptographic lsi with iso/iec 18033-3 block ciphers. In *2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 84–92, Sep. 2009.

[19] M. Fyrbiak, S. Strau, C. Kison, S. Wallat, M. Elson, N. Rummel, and C. Paar. Hardware reverse engineering: Overview and open challenges. In *2017 IEEE 2nd International Verification and Security Workshop (IVSW)*, pages 88–94, July 2017.

[20] Q. Ge, Y. Yarom, D. Cock, and G. Heiser. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *Journal of Cryptographic Engineering*, 8(1):1–27, 2018.

[21] D. Genkin, A. Shamir, and E. Tromer. RSA key extraction via low-bandwidth acoustic cryptanalysis. In *International cryptology conference*, pages 444–461. Springer, 2014.

[22] M. Gruhn and T. Müller. On the Practicability of Cold Boot Attacks. In *2013 International Conference on Availability, Reliability and Security*, pages 390–397, Sep. 2013.

[23] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest We Remember: Cold Boot Attacks on Encryption Keys. pages 45–60, 01 2008.

[24] T. Halevi and N. Saxena. Keyboard acoustic side channel attacks: exploring realistic and security-sensitive scenarios. *International Journal of Information Security*, 14(5):443–456, 2015.

[25] S. Hamdioui, J. Danger, G. Di Natale, F. Smailbegovic, G. van Battum, and M. Tehranipoor. Hacking and protecting ic hardware. In *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1–7, March 2014.

[26] N .Hatami, R. Baranowski, P. Prinetto, and H. Wunderlich. Multilevel simulation of nonfunctional properties by piecewise evaluation. *ACM Trans. Des. Autom. Electron. Syst.*, 19(4):37:1–37:21, August 2014.

[27] M. Hutter and J. Schmidt. The temperature side channel and heating fault attacks. In *International Conference on Smart Card Research and Advanced Applications*, pages 219–235. Springer, 2013.

[28] S. Islam, A. Moghimi, I. Bruhns, M. Krebbel, B. Gulmezoglu, T. Eisenbarth, and B. Sunar. Spoiler: Speculative load hazards boost rowhammer and cache attacks. *arXiv preprint arXiv:1903.00446*, 2019.

[29] P. Kaushik and R. Majumdar. Timing attack analysis on aes on modern processors. In *2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pages 462–465, Sep. 2017.

[30] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping bits in memory without accessing them: An experimental study of dram disturbance errors. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 361–372, June 2014.

[31] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom. Spectre attacks: Exploiting speculative execution. In *40th IEEE Symposium on Security and Privacy (S&P'19)*, 2019.

[32] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Annual International Cryptology Conference*, pages 388–397. Springer, 1999.

[33] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Annual International Cryptology Conference*, pages 104–113. Springer, 1996.

[34] F. Koushanfar. Hardware metering: A survey. In *Introduction to Hardware Security and Trust*, pages 103–122. Springer, 2012.

[35] F. Koushanfar and G. Qu. Hardware metering. In *Proceedings of the 38th Design Automation Conference (IEEE Cat. No.01CH37232)*, pages 490–493, June 2001.

[36] Y. Li, M. Chen, and J. Wang. Introduction to side-channel attacks and fault attacks. In *2016 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC)*, volume 01, pages 573–575, May 2016.

[37] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. Meltdown: Reading kernel memory from user space. In *27th USENIX Security Symposium (USENIX Security 18)*, 2018.

[38] J. Loughry and D. A. Umphress. Information leakage from optical emanations. *ACM Transactions on Information and System Security (TISSEC)*, 5(3):262–289, 2002.

[39] S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks: Revealing the secrets of smart cards*, volume 31. Springer Science & Business Media, 2008.

[40] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld. Physical one-way functions. *Science*, 297(5589):2026–2030, 2002.

[41] J. Quisquater and D. Samyde. Electromagnetic analysis (ema): Measures and counter-measures for smart cards. In *International Conference on Research in Smart Cards*, pages 200–210. Springer, 2001.

[42] S. Subhra R. Chakraborty and S. Bhunia. Hardware protection and authentication through netlist level obfuscation. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-*

Aided Design, pages 674–677. IEEE Press, 2008.

[43] V .Ranade, J. Smith, and B. Switala. Acoustic side channel attack on atm keypads. 2009.

[44] J. Da Rolt, G. Di Natale, M. Flottes, and B. Rouzeyre. A novel differential scan attack on advanced dft structures. *ACM Trans. Des. Autom. Electron. Syst.*, 18(4):58:1–58:22, October 2013.

[45] M. Rostami, F. Koushanfar, and R. Karri. A primer on hardware security: Models, methods, and metrics. *Proceedings of the IEEE*, 102(8):1283–1295, Aug 2014.

[46] J. M. Schmidt and M. Hutter. *Optical and EM fault-attacks on CRT-based RSA: Concrete results*. na, 2007.

[47] M. Schwarz, M. Lipp, D. Moghimi, J. Van Bulck, J. Stecklina, T. Prescher, and D. Gruss. Zombieload: Cross-privilege-boundary data sampling. *arXiv preprint arXiv:1905.05726*, 2019.

[48] M. Schwarz, S. Weiser, D. Gruss, C. Maurice, and S. Mangard. Malware guard extension: Using sgx to conceal cache attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 3–24. Springer, 2017.

[49] M. Seaborn and T. Dullien. Exploiting the dram rowhammer bug to gain kernel privileges. *Black Hat*, 15, 2015.

[50] Q. Shi, H. Wang, N. Asadizanjani, M. M. Tehranipoor, and D. Forte. A comprehensive analysis on vulnerability of active shields to tilted microprobing attacks. In *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pages 98–103, Dec 2018.

[51] S. Skorobogatov. Data remanence in flash memory devices. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 339–353. Springer, 2005.

[52] S. Skorobogatov. How microprobing can attack encrypted memory. In *2017 Euromicro Conference on Digital System Design (DSD)*, pages 244–251, Aug 2017.

[53] P. Socha, J. Brejnk, and M. Bartik. Attacking aes implementations using correlation power analysis on zybo zynq-7000 soc board. In *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4, June 2018.

[54] R. Spreitzer, V. Moonsamy, T. Korak, and S. Mangard. Systematic classification of side-channel attacks: a case study for mobile devices. 2018.

[55] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor. Hardware trojans: Lessons learned after one decade of research. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 22(1):6, 2016.

[56] Y. Xiao, X. Zhang, Y. Zhang, and R. Teodorescu. One bit flips, one cloud flops: Cross-vm row hammer attacks and privilege escalation. In *25th USENIX Security Symposium USENIX Security 16)*, pages 19–35, 2016.

[57] B. Yang, K. Wu, and R. Karri. Scan based side channel attack on dedicated hardware implementations of data encryption standard. In *2004 International Conferce on Test*, pages 339–344, Oct 2004.

[58] Y. Yarom and K. Falkner. Flush+ reload: a high resolution, low noise, l3 cache side-channel attack. In *23rd USENIX Security Symposium USENIX Security 14)*, pages 719–732, 2014.

[59] B. B. Yilmaz, M. Prvulovic, and A. Zaji. Electromagnetic side channel information leakage created by execution of series of instructions in a computer processor. *IEEE Transactions on Information Forensics and Security*, 15:776–789, 2020.