A Low-cost Hardware Attack Detection Solution for IoT Devices

Jordane Lorandel, Mohamed Amine Khelif, Olivier Romain

A Low-cost Hardware Attack Detection Solution for IoT Devices

Conference Paper · June 2022

DOI: 10.1109/ISIE51582.2022.9831661

3 authors:

Jordane Lorandel
Université de Rennes 1 - IETR
27 PUBLICATIONS **129** CITATIONS
SEE PROFILE

Mohamed Amine Khelif
Université de Cergy-Pontoise
7 PUBLICATIONS **25** CITATIONS
SEE PROFILE

Olivier Romain
CY Cergy Paris University
219 PUBLICATIONS **1,823** CITATIONS
SEE PROFILE

A Low-cost Hardware Attack Detection Solution for IoT Devices

Jordane Lorandel[1], Mohamed Amine Khelif[1,2], Olivier Romain[1]
[1]*ETIS Laboratory UMR 8051, CY Cergy Paris University, ENSEA, CNRS, F-95000 Cergy, France*
[2]*ESIEE-IT, 8 rue Pierre de Coubertin, 95300 Pontoise, France*
{jordane.lorandel, mohamed-amine.khelif, olivier.romain}@cyu.fr

***Abstract*—Hardware security becomes a major concern for the development of future IoT devices. In this paper, we propose to investigate traffic monitoring for a low-cost real-time anomaly detection for IoT. This solution relies on the characterization of the wired communication traffic occurring at the interface of the IoT devices, to detect potential attacks and abnormal behaviour by an intrusive solution. A specific hardware architecture is presented and evaluated in an experimental study, showing the effectiveness of the traffic modeling and the detection against two types of hardware attacks performed on the I2C bus.**

***Index Terms*—Security, Hardware Attack, I2C, FPGA, IoT.**

I. Introduction

In the context of the Internet of Things (IoT), a lot of small connected devices is dispersed in the environment enabling the development of smart applications in many domains, starting from smart homes/buildings, e-health, smart farming, and industrial process control, autonomous vehicle, etc. These objects are able to sense, process data depending on their own capability and functionality. In 2020, for the first time, the number of IoT connections has exceeded the number of non-IoT connections (smartphones, laptop) [1], revealing the current massive deployment of IoT devices. Around 31 billion of devices is expected by 2025, which is almost 4 devices per person in the world [1].

However, it has been shown that most of the communications generated by these devices are still not encrypted [2]. More than half of them may reveal major vulnerabilities. In fact, these vulnerabilities could result from the embedded software itself, a misconfiguration, or a hardware weakness. Such vulnerabilities could be potentially exploited by an attacker and in the worst case, the integrity or the confidentiality of user data can be compromised. In some cases, an attacker could take control of the device itself to deploy on top of it a botnet [3]. This idea is presented as the Internet of Vulnerabilities (IoV) in [4].

Starting from this observation, security has become a major concern for all the actors. In fact, the design of IoT devices was preliminary guided to meet typical constraints of embedded systems such as a limited memory capacity, short battery life, and small computing power. The security aspect of these objects appears to be often neglected or secondary.

For the last decade, many attacks have been realized on different target devices, that were categorized as software, hardware, and protocol attacks in [5]. This reveals many remaining challenges to ensure a high level of security. In particular, hardware attacks are very difficult to prevent and could allow an attacker locating sensitive data or taking control of the system. For this type of attack, an assumption is that the attacker has physical access to the device, which seems reasonable regarding the massive deployment of IoT devices. For the future design of these devices, new security solutions have to be thus created with respect to the embedded constraints.

Generic hardware architecture of a typical IoT node is illustrated in Figure 1. As it can be seen, it mainly consists of a sensing unit, a processing part, a radio-frequency transceiver, a power management unit, and a memory. All these elements are interacting together, using different communication standards e.g. I2C, SPI, or UART. This device can be further customized depending on the final application. Such an architecture reveals many possible attack vectors for an attacker including wired communication protocols, wireless connectivity chips, memories, etc.

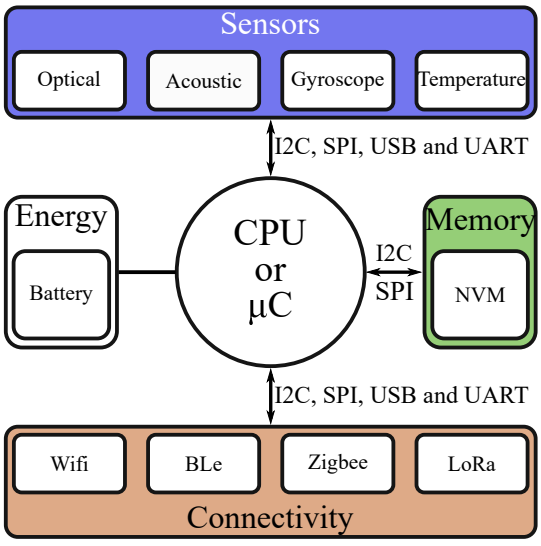

Fig. 1. Simplified IoT hardware device architecture.

In this paper, we focus on the hardware security vulnerabilities targeting widely used communication standards used in many IoT nodes, and more particularly I2C. In fact, this type of communication standard is essential in IoT devices to ensure data exchange between the different chips that compose the IoT device. For instance, the I2C bus could be used to exchange data between the sensing and the processing units or between the processing unit and an external chip. Despite the fact that this technology is very common, it represents a real opportunity for an attacker for gathering particular information about the system, for penetrating the system by creating a fault on the protocol, or by injecting special frames. Moreover, this bus is more vulnerable in comparison to other protocols as an attacker could simply plug his malicious device on the pull up resistors, without being more invasive. In parallel, no specific mechanisms are usually deployed to improve the security of the communication interfaces, in particular when considering small IoT devices based on low-cost micro-controller.

The key contribution of the paper consists in proposing the preliminary element towards the development of an embedded low-footprint security monitoring circuit that allows real-time detection of hardware attacks targeting the wired communication interface used in small IoT devices. To better illustrate the interest of the approach, the solution is evaluated by performing typical hardware attacks on the I2C bus such as frame injection and heart bleed [6].

This paper is organized as follows. First, Section II presents the different categories of hardware attacks with a focus on attacks targeting wired communication buses. Then, the attacks on I2C proposed approach and the low-level implementation details for the real-time attack detection block based on traffic characterization are presented in Sections III and IV respectively. Finally, real experiments of attack on I2C bus are given in Section V and discussed in Section VI before the conclusion.

II. Related Works

From the last decade, many types of attacks have been performed on IoT devices, showing many vulnerabilities and security issues. Among them, hardware attacks represent a particular category of interest as they are very challenging and expensive to counter. However, they are generally more difficult to perform in comparison to software and protocol attacks since they require prior knowledge of the system and its behaviour through a reverse engineering phase [7]. In this section, we focus on hardware attacks and more particularly the ones targeting common wired communication buses.

Hardware attacks include attacks that can be qualified as either passive or active. Considering passive attacks, they require neither opening the chip package nor modifying the IoT device. Instead of that, they consist in measuring the variation of electromagnetic radiation, power consumption, timing execution, temperature, or other metrics of interest from the target device to leak information [8]. Side channel-attack is one of the most used passive attacks.

When considering an active hardware attack, the attacker could choose to perform a hardware modification of the chip or to target an external interface of a communication bus. Several communication protocols represent a potential attack surface. For instance, USB [9], I2C [10], JTAG [11], UART [12], CAN [13] and SPI [14] can be physically accessed from any available pins of the devices without any modification.

The I2C bus is one of the most popular serial communication buses. It is an open protocol that makes serial communication between many peripherals easy. However, it is also simple to disrupt communications within this bus. As presented in [10], the I2C bus attacked using a clock glitching approach. During the attack, the I2C clock line (SCL) is forced to low while communication between a master and a slave peripheral occurs. This allows a malicious attacker to prevent the master to send its message to its selected slave. However, the attacker has to generate acknowledgments to the master instead of the initial target slave in order to not drive the bus to a fault state and to let the master believes that the transaction is effective. As countermeasure, the authors suggest a clock frequency sensor to detect variation in SCL.

In [15], a key-logger attack was demonstrated over the USB. This attack permits to record and transmit all the data that are exchanged over the bus. It also allows to write and modify data. This work shows how it is easy to recover passwords or confidential data using this attack.

Other works have attacked the UART bus with the objective of accessing firmware or sensitive data stored into a component. This information is then used to find and exploit security vulnerabilities that will allow a more complex and potentially damaging attack to be carried out [12]. This is because the UART bus is used in the majority of devices in the IoT as a port for diagnostics and debugging. Also, the UART port is generally poorly protected which makes the attack not very difficult to undertake.

As for the UART bus, the SPI bus is mainly used for attacks that aim to recover data and firmware [14]. The biggest constraint of the SPI bus is that there is no fixed configuration and the number of wires (slave select) depends on the number of components connected to it, except if a daisy chain approach is realized.

The Controller Area Network (CAN) bus is a widely used standard in automotive and industrial applications which allows secure communications with a high level of error detection and a strong electromagnetic protection. However, it is still possible to implement a hardware trojan on this bus

as shown in [16]. In [13], a CAN bus monitoring approach with detection and localization of attacks is proposed as a counter-measure. The solution is based on machine learning and demonstrates a very high accuracy up to 100% for the detection of intrusion and frame injection and 98% for replacement scenarios. However, this approach necessitates to have a large dataset available and seems inappropriate to fit on small IoT devices.

Another type of hardware attack relies on fault injection in order to perturb the chip behaviour and to leak secrets. They require either to be able to interface the target chip in a non-conventional way or the decapsulation of the chip without altering its die. It can be passive if the attack is performed on power and clock lines [17]. More invasive attacks are possible on the depacked device/chip by modifying memory content of RAM, cache, EEPROM, or NVM using laser, UV beams or electromagnetic waves [18]. As demonstrated by [7], not only external interface can be the target of attacks. Indeed, internal communication data buses, even high-performance ones such as PCIe can be interfaced to extract information passively using a sniffer approach or actively by using a Man-in-the-Middle approach.

Despite the panel of possible hardware attacks on serial communication buses, some solutions have already been proposed to enhance IoT device security. For example, a co-processor called SPI-Nooper [19] is in charge of monitoring internal SPI communications between a processor and its radio-frequency transceiver in a Telo mote. It also allows to log traffic, check data integrity and perform abnormal behaviour detection. Other solutions based on cryptography to encrypt data stored into external memory chips could mitigate most hardware attacks. However, cryptography consumes a lot of power and memory resources and is not necessarily used in IoT devices. Moreover, even if the data exchanged on the communication bus are encrypted, the other fields used by the communication protocol are generally not encrypted (target slave address in I2C, the header of PCIe packet, etc.). From this observation, many innovative security solutions could be developed on top of this source of information, essential in the implementation of the communication protocol.

III. Attack on the I2C Interface

A. I2C protocol

I2C is a widely used protocol in the industry for interconnecting multiple elements together using only on two signals, SCL and SDA for clock and signal respectively. Based on a Master/Slave communication scheme, the master always imposes the communication to the different slaves on the bus and is in charge of generating the clock signal, the start of the packet and the end of the packet by driving the lines to low or releasing them to one. A 7-bit/10-bit addressing field allows a master to select a slave while a read/write bit permits to define the type of request [20]. The selected slave can only answer the master with the request data and by sending acknowledgement (ack/nack) bits.

Many attacks such as sniffing, heart bleeding and buffer overflow can be performed on this bus as presented in [6]. In case of heart bleeding attack, the attacker needs to force the ack/nack bit to low in order to avoid the negative acknowledgment that normally stops the memory request. This allows an attacker collecting more information than expected but could also generate an error on the Master side. In addition, an attacker can perform an I2C frame injection to retrieve particular information from the slaves connected to the bus. If the attacker is able to perform a time analysis of the system, malicious frames could be injected without occurring error on the bus or triggering security mechanisms.

B. Discussion and positioning

I2C represents an interesting surface attack vector due to its massive use in IoT devices. Many devices with diverse complexities are concerned going from simple connected lights to the most advanced ones like smartphones or Raspberry pi. Even if I2C is often dedicated to low throughput communications due to its simplicity, it is also widely used as a data bus between a processor or micro-controller and its memories. For instance, Apple's iPhones, iPads, and MACs are currently using the I2C bus between their Secure Enclave chip and an EEPROM memory in which sensitive data such as the password attempt counter and the secure boot are stored [21]. As a consequence, this bus would represent an interesting source of information for an attacker. Developing security solutions that are able to detect an attack or an anomaly in real-time at low cost, remains a challenge, especially when dealing with the low power requirements of very small IoT devices. Even if statistical or machine-learning solutions have proven to be effective on widely used buses [13], there is still a lack of solutions for very small IoT devices.

IV. Real-Time Traffic monitoring and characterization

A. Modeling of I2C traffic

The traffic generated inside a chip or at its interface can provide many information about the behaviour of the system itself. Indeed, in case of attack, such as a botnet, the amount of traffic exchanged on the communication buses would significantly increase in comparison to normal behaviour. It is thus of further importance to be able to characterize the traffic in order to design a solution allowing the detection of a potential attack or anomaly on the device. From this observation, we propose to compute and collect specific metrics to model the behaviour of a traffic at low implementation cost. The defined metrics are computed from specific information obtained from fields related to the communication protocol, such as the number of write requests, number of bytes per frame, etc. We explain this choice because this type of information can be obtained on top of any communication protocol and works even if the

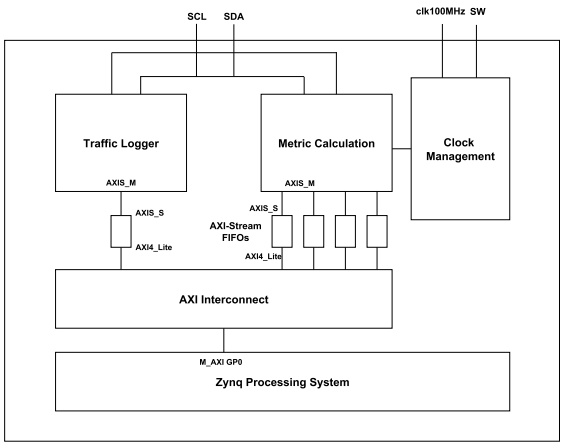

Fig. 2. Hardware Architecture for Traffic logging and Metric Calculation on I2C.

raw data are encrypted.

In our study, the following metrics were defined to characterize the I2C traffic:

- TBF: Time between successive frames
- AT: Time of each active frame
- $NbByte$: Number of exchanged byte
- $NbFrame$: Number of frame
- $NbAckNack$: Number of ACK and NACK bits

In order to develop algorithmic solutions that could exploit these metrics and to make easier the characterization of the traffic over the time, we measure and average them on a user-defined time window. Note that the raw data can also be logged and processed offline in order to retrieve additional information such as the number of slaves, number access per slave, number of R/W requests, etc. This only necessitates a sniffer device or logic analyzer. Many solutions are available off-the-shelf such as Analog Discovery 2 Kit from Avnet. In addition, one can note that the considered metrics are voluntarily kept as generic as possible in order to let the possibility of characterizing many other wired communication protocols such as SPI, CAN, etc. In addition, bus-specific metrics could also be easily integrated.

B. Low-level Hardware Architecture

To compute the metrics defined in the previous sub-section, a hardware IP was designed. The low-level details about the hardware implementation are given by Figure 2.

The solution relies on an I2C slave interface that allows logging all the traffic exchanged on the bus without perturbing the communication. To this purpose, SCL/SDA input pins are configured in high-state mode. As it can be seen, the I2C lines are then sent to two modules, a traffic logger and a metric calculation core. For these 2 IPs, external AXI-Stream interfaces were chosen to permit AXI4-compliant connections with other standard IPs and a fast design development. For instance, the outputs of these IPs are connected to AXI-Stream FIFOs from Xilinx. These FIFOs allows to transfer the logged data and the computed metrics to the processor through AXI4-Lite Interfaces. Regarding the software design, the processor is polling the empty signal of the different FIFOs to get both the raw data or the metrics. The processor could either output the metrics through a UART to an external PC or also performs some additional processing directly on the metrics if needed.

From a practical point of view, the interfacing of this solution to the I2C bus under characterization could be easily performed by soldering wires on the pull-up resistors of the target device, after a reverse engineering phase to retrieve their localization. This also demonstrates that an independent and external traffic characterization solution could be added to a small IoT device without redesigning or perturbing it.

Another particularity of the solution is the possibility to select the sampling frequency of the metrics, respectively the duration of the time window. A 2-bit signal named (SW) allows the selection of the duration between $1ms$, $0.01s$, $0.1s$ and $1s$. A typical clock of 100MHz has to be provided at the corresponding input to generate the multiple clock frequencies.

V. Traffic Modeling and Detection of Attack

To validate our solution, we developed an experimental set-up based on an MCP9808 sensor, a Nucleo-F446RE and an E2PROM memory, all interconnected through an I2C bus.

The application running on the Nucleo-F446RE, acting as Master, consists in getting the temperature from the MCP9808 every second and then writing the results to the external E2PROM memory. The corresponding setup is illustrated in Figure 3.

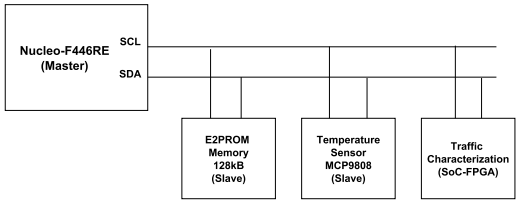

Fig. 3. Validation setup for the I2C Traffic Analysis.

The solution for traffic characterization was implemented onto a Zedboard platform from Avnet [22], which includes a Zynq-based SoC device, combining an FPGA-based programmable logic (PL) and a dual-core Cortex-A9 processor as the programmable system (PS). In our case, the PL side implements the IP while the PS is used to control the logic, configure the time window and send the results of the traffic analysis to an external PC through UART.

The implementation resources used by the IP are summarized in Table I. As it can be seen, the overall solution

TABLE I
HARDWARE RESOURCE UTILIZATION ON A ZEDBOARD PLATFORM.

Description	LUT	Flip-Flop	BRAMS
Metric Calculation + clock Management	398	654	0
Traffic Logger	60	83	0
AXI-Stream FIFOs (x5)	362	357	1
Total (%)	7	5	5

is using very few logic and memory resources with respect to the FPGA device available on the Zedboard platform, respectively 7% of LUT, 5% of Flip-Flops and 5 BRAMS. Note that 5 AXI-Stream FIFOs are required in total for collecting both raw data and the defined metrics.

During this experimental setup, we recorded the considered metrics for several minutes and for different scenarios. In the first scenario, no attack is generated on the system. The expected behaviour is the normal one, regarding the metrics. For the second scenario, we developed another architecture onto an FPGA device, to perform a frame injection attack on the I2C bus. Two malicious frames will be injected at different times without generating any error on the bus or perturbing the application. In the third scenario, a heart bleeding attack was realized in a similar way as in [6].

Figure 4 gives the corresponding histograms obtained for the TBF (Time Between Frame) metric, considering an analysis time of around 2 minutes for each scenario. As it can be seen, TBF metric provides enough information for detecting an abnormal behaviour of the system with respect to the 2 attacks performed. For instance, we observe a dispersion of the TBF values due to the frame injection attack in comparison to the two distinct areas in the normal behaviour.

Regarding the heart bleeding attack, the number of TBF values close to 0 significantly increases in comparison to the normal case. This is expected because, for this type of attack, the communication between the master and its slave is altered. Indeed, the ack/nack bit of the master is forced to low by the attacker, meaning that the target transaction is not finished and the slave (the E2PROM memory in our case) keeps sending packets to the attacker that maintains the communication by generating a clock on the SCL line. In our solution, the metric calculation core does not update the value of the TBF as the end of the transaction does not appear. Recall that this type of attack allows an attacker to retrieve more information as needed, for example, the content of additional memory addresses.

Regarding another metric such as the measurement of the duration of valid I2C frames (expressed in clock cycles), the detection of the attack can also be observed as illustrated by Figure 5. For a frame injection attack, the histogram reveals the emergence of additional frame duration due to the new

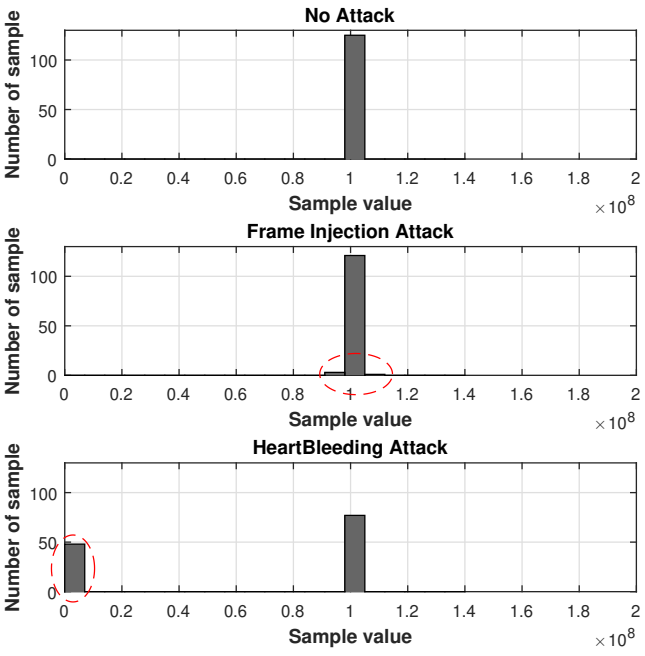

Fig. 4. Histogram of the $time\ between\ frame$ metric for the 3 scenarios.

frames injected by the attacker. The detection of the heart bleeding attack is also visible, as for the previous metric, with the apparition of many values close to 0. The same conclusion could be drawn when studying complementary metrics offered by the design, such as $NbByte$, $NBFrame$ and $NbAckNack$.

VI. DISCUSSION AND ANALYSIS

The characterization of the traffic seems well adapted for systems that exhibit periodic tasks. We believe that most of the applications targeting small IoT devices are dedicated to periodic sensing processing and transmission of data. The traffic generated on the communication buses should remain roughly the same, even if the periodicity is changed, making this solution well adapted.

As previously mentioned, other solutions for attack detection exist such as evolved statistical and machine learning techniques [13] but they still remain very complex to realize and require a high computing power that is not necessarily available on small IoT devices. The prototype made in this project was firstly developed on FPGA but the used resources could be significantly optimized with an ASIC implementation.

The solution proposed in this paper corresponds to the first element of the security block with the detection of the attack. A further analysis would be the classification and/or identification of the type of attack by studying the behaviour of the defined metrics.

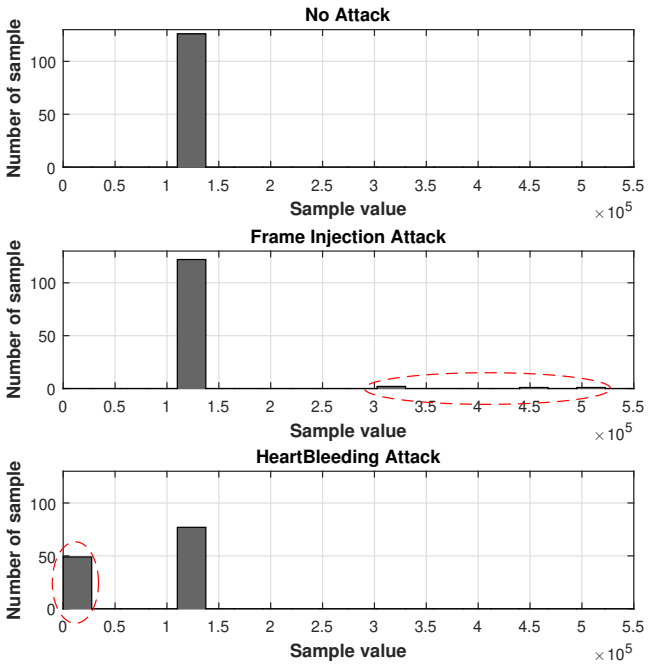

Fig. 5. Histogram of the *Active Time* metric for the 3 scenarios.

Moreover, as many built-in sensors are available in small IoT devices, an interesting approach would be to jointly combine the recording of metrics related to the communication part with non-functional ones e.g. the device's temperature or power consumption.

Another interesting opportunity is the use of low-complexity algorithms in combination with this hardware design to develop a real-time anomaly detection solution. In particular, online anomaly detection algorithms have proven to perform well as online estimators [23] against DDoS attacks. The proposed design would be the first element of a self-monitoring security solution that will assess an abnormal behaviour in real-time and low-cost.

VII. Conclusion

In this paper, we present the first element toward the design of low-cost real-time anomaly detection for IoT devices. The solution relies on the characterization of the wired communication traffic occurring at the interface of the IoT devices, to detect potential attacks. A specific hardware architecture is presented, showing the low footprint of the solution which makes it compatible to be embedded with small IoT devices. A successful detection of two hardware attacks targeting the widely used I2C bus was demonstrated. The future works will focus on demonstrating the genericity of the approach on other IoT communication buses in combination with online estimators.

References

[1] K. L. Lueth, "State of the iot 2020: 12 billion iot connections, surpassing non-iot for the first time," 2020. [Online]. Available: https://iot-analytics.com/

[2] O. L., "More than half of iot devices vulnerable to severe attacks," 2020. [Online]. Available: https://threatpost.com/half-iot-devices-vulnerable-severe-attacks/153609/

[3] D. McMillen, W. Gao, and C. DeBeck, "A new botnet attack just mozied into town," 2021. [Online]. Available: https://securityintelligence.com/posts/botnet-attack-mozi-mozied-into-town/

[4] K. Angrishi, "Turning internet of things(iot) into internet of vulnerabilities (iov) : Iot botnets," *CoRR*, vol. abs/1702.03681, 2017. [Online]. Available: http://arxiv.org/abs/1702.03681

[5] M. A. Khelif, J. Lorandel, O. Romain, M. Regnery, D. Baheux, and G. Barbu, "Toward a hardware man-in-the-middle attack on pcie bus," *Microprocessors and Microsystems*, vol. 77, p. 103198, 2020.

[6] M. A. Khelif, J. Lorandel, and O. Romain, "Non-invasive i2c hardware trojan attack vector," in *2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2021, pp. 1–6.

[7] M. A. Khelif, J. Lorandel, O. Romain, M. Regnery, D. Baheux, and G. Barbu, "Toward a hardware man-in-the-middle attack on pcie bus for smart data replay," in *2019 22nd Euromicro Conference on Digital System Design (DSD)*. IEEE, 2019, pp. 230–237.

[8] G. Joy Persial, M. Prabhu, and R. Shanmugalakshmi, "Side channel attack-survey," *Int J Adva Sci Res Rev*, vol. 1, no. 4, pp. 54–57, 2011.

[9] B. Lau, Y. Jang, C. Song, T. Wang, P. H. Chung, and P. Royal, "Mactans: Injecting malware into ios devices via malicious chargers," *Black Hat USA*, 2013.

[10] F. Gomez-Bravo, R. J. Naharro, J. M. García, J. G. Galán, and M. Raya, "Hardware attacks on mobile robots: I2c clock attacking," in *Robot 2015: Second Iberian Robotics Conference*. Springer, 2016, pp. 147–159.

[11] G. Vishwakarma and W. Lee, "Exploiting jtag and its mitigation in iot: a survey," *Future Internet*, vol. 10, no. 12, p. 121, 2018.

[12] A. Gupta, *The IoT Hacker's Handbook*. Springer, 2019.

[13] E. Levy, A. Shabtai, B. Groza, P. Murvay, and Y. Elovici, "CAN-LOC: spoofing detection and physical intrusion localization on an in-vehicle CAN bus based on deep features of voltage signals," *CoRR*, vol. abs/2106.07895, 2021. [Online]. Available: https://arxiv.org/abs/2106.07895

[14] Asmita-Jha, Payatu, "IoT Security - Part 15 (101 - Hardware Attack Surface : SPI)," https://bit.ly/3iThQTo, 2020, accessed: 02/05/2021.

[15] P. D. Bojovic, I. Basicevic, M. Pilipovic, Z. Bojovic, and M. Bojovic, "The rising threat of hardware attacks: Usb keyboard attack case study," *Preprint*, 2019.

[16] M. Bozdal, M. Randa, M. Samie, and I. Jennions, "Hardware trojan enabled denial of service attack on can bus," *Procedia Manufacturing*, vol. 16, pp. 47–52, 2018.

[17] H. Martin, T. Korak, E. San Millán, and M. Hutter, "Fault attacks on strngs: Impact of glitches, temperature, and underpowering on randomness," *IEEE transactions on information forensics and security*, vol. 10, no. 2, pp. 266–277, 2014.

[18] M. Eslami, B. Ghavami, M. Raji, and A. Mahani, "A survey on fault injection methods of digital integrated circuits," *Integration*, vol. 71, pp. 154–163, 2020.

[19] M. S. Hossain, W. S. Lee, and V. Raghunathan, "Spi-snooper: a hardware-software approach for transparent network monitoring in wireless sensor networks," in *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, New York, NY, USA, 2012, pp. 53–62. [Online]. Available: http://dl.acm.org/citation.cfm?id=2380460

[20] J.-M. Irazabal and S. Blozis, ""i2c manual" application note," NXP Semiconductors, Tech. Rep., 2003.

[21] Apple, "ios security ios 12.1," Apple Inc, Tech. Rep., 2018.

[22] Avnet, "Zedboard zynq-7000 platform," 2021. [Online]. Available: https://www.avnet.com/

[23] S. Skaperas, L. Mamatas, and A. Chorti, "Real-time video content popularity detection based on mean change point analysis," *IEEE Access*, vol. 7, pp. 142 246–142 260, 2019.