End Node Security and Trust vulnerabilities in the Smart City Infrastructure

Apostolos P. Fournaris[1], *Konstantinos* Lampropoulos[1], *Odysseas* Koufopavlou[1]

[1]Electrical and Computer Engineering Dpt, University of Patras, Rion Campus, Greece

Abstract. As cities gradually introduce intelligence in their core services and infrastructure thus becoming "smart cities", they are deploying new Information Technology devices in the urban grid that are interconnected to a broad network. The main focus of widely implemented smart cities' services was the operation of sensors and smart devices across city areas that need low energy consumption and high connectivity. However, as 5G technologies are gradually been adopted in the smart city infrastructure thus solving that problem, the fundamental issue of addressing security becomes dominant. While latest network topologies and standards include security functions thus giving an illusion of security, there is little focus on the fact that many smart city end nodes cannot realize all security specifications without additional help. In this paper, we discuss briefly smart city security issues and focus on problem and security requirement that need to be address in the smart city end nodes, the sensors and actuators deployed within the city's grid. In this paper, attacks that cannot be thwarted by traditional cybersecurity solutions are discussed and countermeasures based on hardware are suggested in order to achieve a high level of trust. Also, the danger of microarchitectural and side channel attacks on these devices is highlighted and protection approaches are discussed.

1 Introduction

Many cities around the globe have started becoming "smart cities", deploying various technologies and digital infrastructures to increase the quality of their services and consequently the quality of life of their citizens. Smart cities offer wide opportunities by concentrating urban services and fostering intelligence within networks. A smart city collects various types of electronic data and processes them to a) manage its assets and resources efficiently and b) improve the operations and services provided to its citizens. Data are generally collected from citizens, devices, assets etc. and are used for the optimization of transportation systems, power plants, water supply networks, waste management, law enforcement, public safety, etc.

Until recently, data collection for smart cities applications was not an easy task. The main problem was that the operation of sensors and smart devices across wide city areas was facing problems in a) energy consumption and b) connectivity. In particular, smart devices/sensors could not operate more than a few days without a power source and available network protocols (WiFi, 4G) could not provide efficient connectivity because

WI-FI was designed for low range applications and 4G required a business contract with a Telecommunication company.

These problems are now being addressed with the new 5G technologies. Protocols like LoRa[*] NB-IoT[†], Zigbee[‡], Sigfox[§] etc can support low power, wide area communications, and allow the deployment of nodes in remote or difficult to access areas. Industrial version of such nodes can now operate for a duration between 5 to 10 years with the use of only batteries[**]. At network layer the above end node technologies adopt the IP network model thus remaining fully compatible with traditional Internet network resources. Also, among the standardized solutions, using nodes that communicate under a low power MAC/Physical layer infrastructure (different than the power hungry WiFi, Ethernet), like IEEE 802.15.4 [1], IETF has standardized the CoAP protocol stack based on 6lowPAN (reduced IP based) network protocol in an effort to provide HTTP-like connectivity for the IoT environment [2][3].

But despite the fact, that smart cities are growing very fast, security is still a big issue for them. In addition to traditional computer/networks threats, IoT systems are also vulnerable to physical attacks since sensors can be deployed in areas with public access or isolated with minimum supervision. At the same time various deployments introduce complete solutions that combine the communication and application process in one single product (The Things Network - LoRa), leaving the overall framework open to multiple types of attacks [4]. Standardization activities, like those performed by Zigbee alliance or IETF are gradually focusing on security but they are still not providing application layer security (eg. end-to-end security, authorization and authentication services).

Most importantly, however, there is a part of the smart city infrastructure that is overlooked in terms of security. While network is protected, the actual end nodes, having strict requirements in term of performance, power consumption and memory resources (in the case of embedded, cyberphysical end nodes) cannot always support all the network prescribed functionality. Furthermore, they are vulnerable to a broad range of device-based attacks that are not typically taken into account by a smart city IoT architect. Thus, there are still several vulnerabilities in smart cities technologies and services that leave the smart city infrastructure open to a large variety of physical and cyber-attacks.

Discussing all the security issues of smart cities technologies and architectures is not feasible within the context of one single document. Thus, this paper will provide a very abstract description of the infrastructure and architecture of a smart city, discuss the major security aspects of the various architectural layers and finally focus on security issue of the smart city end node devices. Our aim is to highlight the need to protect the device itself from non traditional attacks (like side channel and microarchitectural attacks) and provide some countermeasure that can be used on existing and future device to enhance their security level.

The rest of the paper is organized as follows. In Section 2 an overview of the smart city infrastructure is presented and discussed. In Section 3, an overview of smart city security issues is made discussing possible problems in the various smart city layers. In Section 4 the security of the end nodes is discussed, attacks and countermeasures are presented and Section 5 concludes the paper.

[*] https://www.lora-alliance.org/
[†] https://en.wikipedia.org/wiki/NarrowBand_IOT
[‡] http://www.zigbee.org/
[§] https://www.sigfox.com/en
[**] https://www.worldsensing.com/product/loadsensing-2/

2 Smart City Infrastructure

A smart city uses information and communication technologies (ICT) to collect information from its citizens and devices and process it to improve its operational efficiency, communication with the public and quality of government services and citizen welfare. An abstract design of such a city includes services like smart parking, smart lighting, smart buildings, efficient energy-waste-water etc. management, improved healthcare etc. and it is depicted abstractly in Figure 1.

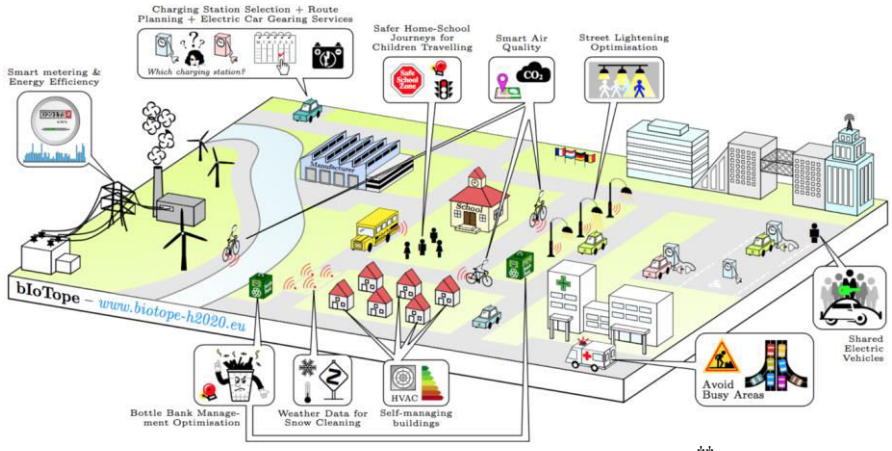

Figure 1. A smart city architecture abstract view[††]

From the technical point of view, the architecture of a smart city is composed of different technologies, services, devices, actors etc. There are various different approaches presenting how these components are connected and orchestrated. In this paper and for the sake of simplicity we will adopt the four layered architecture depicted in Figure 2. This architecture divides the various components of a smart city in the following layers (bottom to top): **Sensing and Control Layer, Communication Layer, Processing Layer, Application Layer**.

Figure 2. Smart city infrastructure various Layers

It is clear that the recent ICT and IoT advances in all four layers played a significant role in creating the technologies and systems to support such an architecture. However, the adaptation of these advances in city services and operations was done rather quickly and

[††] http://ramonelu.info/carn/a/biotope-research/

without proper evaluation on its security impact. In a 2016 report, "Cyber security, A necessary pillar of Smart Cities"[‡‡], authors agree on the fact that the risk landscape is consequently the widest with security and privacy concerns led by insecure hardware, a larger attack surface, bandwidth consumption issues, application risks. Sensitive data and the network scope in smart cities implies high stakes in security to ensure confidentiality, integrity and availability as simple bugs may hinders huge impacts.

3 Overviewing Smart Cities' Security Issues

In view of Figure 2, it is made clear that smart cities are vulnerable to nearly every type of attack in the ICT sector. For the application layer, smart cities applications and services have to deal with injection attacks, cross site scripting, broken authentication/authorization mechanisms leading to authorized access and sensitive information leaking, social engineering, insecure 3rd party applications/components etc. For the processing layer, the attacks include DDoS attacks, hacking and intrusion, worms, viruses and malwares etc. We need to point out that the distinction between the application layer and the processing layer is not always clear (e.g. applications may or may not use their own servers, resources etc.) thus many of the above attacks apply in both layers. For the communication layer smart cities are also facing the attacks of existing network infrastructures. Such attacks include jamming, spoofing, wormholes, man-in-the middle, sinkhole, sybil, eavesdropping, replay etc as those can be manifested in the various layers of the OSI network model. However, most of the above attacks can be mitigated using solutions and products from the Information Technology domain.

A smart city can be seen as a collection of diverse systems forming dynamic applications and services. Thus, complete security cannot be applied in the form of one single framework or product that covers everything. The approach to secure smart cities infrastructures is to a) ensure that its components maintain high levels of security and b) evaluate the vulnerabilities of each new service or application, also examining their security impact on shared systems and resources.

4 End Node Security Issues And Proposed Solutions

The end node of a smart city infrastructure is usually associated with the sensing and control layer of a smart city and partially with the communication layer. End nodes are usually embedded systems devices for sensing or actuating control loops of the overall smart functionality assigned by the city IoT cloud environment administrator. Apart from that, end nodes may also be full Personal computers or workstations that act as control and visualization terminals of the city's overall "health" and are managed by a city's administration authorities (e.g city municipality technicians). The two types of end nodes have completely different characteristics and due to their different roles in the smart city infrastructure have different security needs.

Cybersecurity attacks on the end nodes can assume different forms depending on the kind of end node devices (embedded or Personal Computer). While there exist a broad range of attacks targeting PC devices, widely explored, and thwarted by international literature works and products, the embedded system domain is mostly unexplored (and unprotected) regarding cyberattacks. Many of PC end nodes has countermeasures that effectively

[‡‡] EY, Publications, http://www.ey.com/Publication/vwLUAssets/ey-cyber-security-a-necessary-pillar-of-smart-cities/$FILE/ey-cyber-security-a-necessary-pillar-of-smart-cities.pdf

provide protection against IT based cyberattacks including antimalware programs, firewalls, Intrusion Detection Systems and Anomaly detection tools. Recent attacks that exploit cyber-physical systems have triggered interest in cyberphysical/embedded system cybersecurity countermeasures that yet still adopt the same principles as the PC based ones. However, while most highend software cyber security solution can effectively protect against attacks based on software vulnerabilities, when it comes to hardware vulnerabilities the attackers still have a rich unexplored area to exploit with few countermeasures (if any) to thwart them [5].

A very realistic attack risk on embedded end node devices is associated with the low level of security that such nodes adopt. Typical sensor/actuator solutions deployed in a smart city environment bare no or minimal security features focused on achieving confidentiality through encryption (typically AES 128 block cipher algorithm is used in CBC, CTR mode). Security functions employ pre-shared keys (that are hardly updated after deployment) and no authentication or integrity mechanism. The main reason for the reduced security level in such nodes is associated with the low number of hardware resources, the need for low power consumption and the small processing power of the nodes but also lack of deep security knowledge by the device manufacturers. Attackers gradually have started taking advantage of embedded end nodes low security and have exploited it to hijack IoT end nodes in order to manipulate data or even use IoT end node devices as zombie machines for secondary attacks with devastating effects (eg. The Mirai botnet) [6].

An obvious solution to the above issue is the introduction on each end node, of high quality security features like authenticated encryption modes, key agreement schemes, certificates etc. However, on many low to mid-range embedded system end nodes this is not possible due to low performance capabilities and the need for low power consumption. In order to be able to handle that kind of problems and to achieve a level of trust on the functionality of an end node, the node's security must be supported through hardware means [7][8]. For this reason, modern embedded system processors (eg. ARM based) include some security element (a kind of security primitive Hardware accelerator/coprocessor) in their architecture. In some cases, this functionality is enhanced to provide secure execution functionality on specified software applications. In the smart city domain, such end node applications can be associated with sensor data collection and/or control actuation loops.

On the other hand, the fact that many end nodes, are deployed on unprotected, unsupervised physical environment (in waste bins, on lamp posts, on pavements etc.) they can become targets to physical tampering. This kind of tampering may not involve actual device dismantling but can be a very subtle remote or physical injection of faults or physical characteristic leakage (Electromagnetic emission, power consumption, timing) collection during security function execution [9]. Those types of attacks, known as fault injection and side channel analysis attacks have been proven successful regardless of the security algorithm's strength since they rely on the algorithm's vulnerable implementation and the device physical characteristics.

4.1 Introducing Trust to smart city end nodes

One of the most potent means of enhancing computer security even on low end performance devices is to migrate security functions in hardware. For this reason, in the embedded system world there exist several hardware security elements along with appropriate software libraries that advertise security and trust. To instil trust on a computing system a trusted computing base (TCB) must be introduced to it. A TBC acts as a root of trust, a trusted point of reference for the overall system. TBCs can guarantee (or not) the trustiness of a software component and cannot be tampered. The dominant approach to achieve that is to include a hardware component (a security element) in a

device's architecture capable of handling security services in a secure environment. There are several approaches on how to provide such a hardware root of trust mechanism both industrial and research based. Among the most important ones are the specifications provided by Trusted Computing Group (TCG) aiming to enforce trust on a system by prohibiting the execution of malicious code, by protecting sensitive data (mainly private keys), and by attesting the system's trust level to other entities. This is achieved by a constant evaluation of the computer system's security from boot time. Since software is impossible to remain un-tampered in order to form a TCB, TCG solution is based on hardware protection mechanisms in the form of dedicated element denoted as Trusted Platform Module, along with software tools to establish trust. The TCG's TPM acts as trust anchor within a computer system [10] [11] [12]. The overall concept adopted in the TCG solution, and expanded, differentiated in more processor-oriented approaches is to achieve a level of isolation on the security critical operations on the smart sensor/actuator end node. Following the above directives, as described in [11], there are two approaches that may be introduced to a smart city end node to achieve trust. The first technology to be used is related to a lightweight version of TCG's TPM specification like the Device Identifier Composition Engine (DICE) proposal, which is ideal for Low energy and Low resources devices without a TPM (most commercial embedded systems do not yet have a TPM chip). The second technology is related to the introduction, inside the end node execution path, of a virtual, isolated, environment that is controlled by a processor supporting TCBs. Such technology, acting as a lightweight hardware virtualization technique, can be applied to ARM based processor end nodes where the ARM TrustZone technology is supported, offering isolation between trusted and not trusted execution (trust vs normal zone) [13].

4.2 Microarchitectural Attacks and Side Channel analysis

While the above solutions seem to be able to guarantee a high level of security and trust, there are still things that an attacker can exploit to compromise an end node. Microarchitectural attacks, fault injection and side channel analysis attacks exploiting hardware vulnerabilities are not thwarted by most software countermeasure and can be used in order to bypass even hardware virtualization (eg. Supported by ARM TrustZone).
Microarchitectural attacks [5], [14], [15] exploit the actual computer architecture structure associated with the processor's cache memory, assembly command or thread execution sequence, pipelining, hyperthreading but also focus on technology exploits like DRAM disturbance errors. Such attacks are closely related with side channel information leakage analysis attack and can be viewed as a special case of the later.
Cache attacks are a very popular type of Microarchitectural attacks. They aim at recovering secret keys when executing a known cryptographic algorithm. There exist access based cache attacks where an attacker extracts results from observing his own cache channel (measuring cache access time) and timing based cache attacks where the attacker extracts results from measuring a victim's cache hits and misses to infer access time [5]. Among them worth mentioning are the PRIME + PROBE and EVICT + TIME attack approaches introduced in [16]. Both attacks relied on the knowledge of a cache state before an encryption and the capturing of the cache changes during encryption to deduce sensitive data. After the proposal of the FLUSH + RELOAD attack, described in [17], cache attacks became more potent by exploiting the shared memory pages of OS libraries stored in the Last Level Cache (LLC) of any computer and similarly to sophisticated variations of the PRIME+PROBE attack [18] also focused on LLC, became applicable in cross core applications even against VM devices [19], [20]. This latest development had serious implication about the security of ARM TrustZone and indeed several research work indicate that ARM TrustZone can be compromised using cache attacks [21]–[23]. Latest

exploit on the combination of various microarchitectural vulnerabilities including cache attacks is the Meltdown and Spectre vulnerabilities [24], [25].
DRAM disturbance can also lead to serious exploits. The strange case of RowHammer bug where repeated access of the same memory row can cause enough disturbance in a neighbouring row (victim row) to make a bit flip, has created considerable problems to computer designers since it can be used for a fault injection attack that can be mounted remotely [26], [27]. Researchers have also shown that the problem is evident and exploitable in the embedded world [28] while it can also be used to break VM (and virtualization in general) isolation [29], [30].
Countermeasures against the above attacks are focused on disallowing certain actions related to the computer architecture from the OS userspace (eg. Cache eviction mechanism), on upgrading DRAM memory modules to DDR4 with Error Correcting Code (ECC) that is considered harder to be exploited by Rowhammer attacks and introduce randomness in the cache memory management mechanisms [31]. However, is should be pointed out that most of the countermeasures are patches that mitigate not solve the vulnerabilities. Attackers that can invest more effort and budget may bypass most of the countermeasures. It is assumed that in the case of smart city end nodes an attacker won't be interested to invest that time and effort.

6 Conclusions

As it can be concluded in this paper, security of the end nodes has several peculiarities that cannot be overlooked. End node strong security cannot be achieved without hardware support either by the processor itself or be additional security elements. End nodes can be attacked in an unconventional way by exploiting hardware vulnerabilities that are hard to thwart. Thus, a smart city designer and security administrator should be very careful on the choice of deployed end nodes on the smart city urban grid if he is to retain the security level that latest network standards offer him. Else, the smart city end nodes can be easily used as an attack vector to compromise effortlessly the overall smart city infrastructure.

This work is funded by the SMESEC "Cybersecurity for small and medium-sized enterprises" EU Horizon 2020 project under grant agreement No 740787

References

[1] D. De Guglielmo, S. Brienza, and G. Anastasi, "IEEE 802.15.4e: A survey," *Comput. Commun.*, vol. 88, pp. 1–24, Aug. 2016.

[2] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," Jun. 2014.

[3] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler, "Standardized Protocol Stack for the Internet of (Important) Things," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.

[4] E. Aras, G. S. Ramachandran, P. Lawrence, and D. Hughes, "Exploring the Security Vulnerabilities of LoRa," in *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, 2017, pp. 1–6.

[5] A. Fournaris, L. Pocero Fraile, and O. Koufopavlou, "Exploiting Hardware Vulnerabilities to Attack Embedded System Devices: a Survey of Potent Microarchitectural Attacks," *Electronics*, vol. 6, no. 3, p. 52, Jul. 2017.

[6] M. Antonakakis, T. April, M. Bailey, E. Bursztein, J. Cochran, Z. Durumeric, J. Alex Halderman, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai Botnet," *Proc. 26th USENIX Secur. Symp.*, pp. 1093–1110, 2017.

[7] A. P. Fournaris and D. M. Hein, "Trust Management Through Hardware Means: Design Concerns and Optimizations," in *VLSI 2010 Annual Symposium*, vol. 105, N. Voros, A. Mukherjee, N. Sklavos, K.

Masselos, and M. Huebner, Eds. Springer Netherlands, 2011, pp. 31–45.

[8] R. B. Aussel J D A Sailer, "Only hardware-assisted protection can deliver durable secure foundations," *IEEE Softw.*, vol. 28, no. 2, p. 57+58-59, 2011.

[9] S. Mangard, E. Oswald, and T. Popp, "Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)," Feb. 2007.

[10] T. C. Group, "TPM v2.0 Library Specification," 2014.

[11] A. Fournaris, K. Lampropoulos, and O. G. Koufopavlou, "Trusted Hardware Sensors for Anomaly Detection in Critical Infrastructure Systems," in *in proc. of the 7th International Conference on Modern Circuits and Systems Technologies (MOCAST) on Electronics and Commuications 2018*, 2018.

[12] D. Challener, K. Yoder, R. Catherman, D. Safford, and L. Van Doorn, *A practical guide to trusted computing*. IBM press, 2007.

[13] T. Alves and D. Felton, "TrustZone: Integrated hardware and software security," *ARM white Pap.*, 2004.

[14] Q. Ge, Y. Yarom, D. Cock, and G. Heiser, "A survey of microarchitectural timing attacks and countermeasures on contemporary hardware," *J. Cryptogr. Eng.*, pp. 1–27, 2016.

[15] D. Gruss, "Software-based Microarchitectural Attacks — PhD Defense," no. June, 2017.

[16] D. A. Osvik, A. Shamir, and E. Tromer, "Cache Attacks and Countermeasures: The Case of AES," in *Proceedings of the 2006 The Cryptographers' Track at the RSA conference on Topics in Cryptology*, Springer-Verlag, 2006, pp. 1–20.

[17] Y. Yarom and K. Falkner, "Flush + Reload : a High Resolution , Low Noise , L3 Cache Side-Channel Attack," *USENIX Secur. 2014*, pp. 1–14, 2014.

[18] F. Liu, Y. Yarom, Q. Ge, G. Heiser, and R. B. Lee, "Last-level cache side-channel attacks are practical," *Proc. - IEEE Symp. Secur. Priv.*, vol. 2015–July, pp. 605–622, 2015.

[19] G. Irazoqui, M. S. Inci, T. Eisenbarth, and B. Sunar, "Wait a minute! A fast, cross-VM attack on AES," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8688 LNCS, no. Vmm, pp. 299–319, 2014.

[20] G. Irazoqui, T. Eisenbarth, and B. Sunar, "Cross Processor Cache Attacks," *Proc. 2016 ACM Asia Conf. Comput. Commun. Secur.*, pp. 353–364, 2016.

[21] N. Zhang, K. Sun, D. Shands, W. Lou, and Y. T. Hou, "TruSpy : Cache Side-Channel Information Leakage from the Secure World on ARM Devices," *Cryptol. ePrint Arch. Rep. 2016/980*, 2016.

[22] M. Lipp, D. Gruss, R. Spreitzer, C. Maurice, and S. Mangard, "ARMageddon: Cache Attacks on Mobile Devices," in *Proceedings of the 25th USENIX Security Symposium, pp. 549–564, Austin, TX, US*, 2016.

[23] X. Zhang, "Return-Oriented Flush-Reload Side Channels on ARM and Their Implications for Android Security," *CCS '16 Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Secur.*, pp. 858–870, 2016.

[24] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre Attacks: Exploiting Speculative Execution," Jan. 2018.

[25] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown," Jan. 2018.

[26] S. Bhattacharya and D. Mukhopadhyay, "Curious case of Rowhammer: Flipping secret exponent bits using timing analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9813, pp. 602–624, 2016.

[27] D. Gruss, C. Maurice, and S. Mangard, "Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript," 2015.

[28] V. Van Der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi, and C. Giuffrida, "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms," in *CCS*, 2016.

[29] E. Bosman, K. Razavi, H. Bos, and C. Giuffrida, "Dedup Est Machina: Memory Deduplication as an Advanced Exploitation Vector," *Proc. - 2016 IEEE Symp. Secur. Privacy, SP 2016*, pp. 987–1004, 2016.

[30] M. Seaborn and T. Dullien, "Exploiting the DRAM rowhammer bug to gain kernel privileges How to cause and exploit single bit errors Mark Seaborn and Thomas Dullien Bit flips !"

[31] T. Zhang and R. B. Lee, "Secure Cache Modeling for Measuring Side-channel Leakage," pp. 1–27.