Are Cloud FPGAs Really Vulnerable to Power Analysis Attacks?

Ognjen Glamočanin*, Louis Coulon*, Francesco Regazzoni† and Mirjana Stojilović*
*École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
†ALaRI, Università della Svizzera italiana, Lugano, Switzerland
Email: ognjen.glamocanin@epfl.ch, mirjana.stojilovic@epfl.ch

***Abstract*—Recent works have demonstrated the possibility of extracting secrets from a cryptographic core running on an FPGA by means of remote power analysis attacks. To mount these attacks, an adversary implements a voltage fluctuation sensor in the FPGA logic, records the power consumption of the target cryptographic core, and recovers the secret key by running a power analysis attack on the recorded traces. Despite showing that the power analysis could also be performed without physical access to the cryptographic core, these works were mostly carried out on dedicated FPGA boards in a controlled environment, leaving open the question about the possibility to successfully mount these attacks on a real system deployed in the cloud. In this paper, we demonstrate, for the first time, a successful key recovery attack on an AES cryptographic accelerator running on an Amazon EC2 F1 instance. We collect the power traces using a delay-line based voltage drop sensor, adapted to the Xilinx Virtex Ultrascale+ architecture used on Amazon EC2 F1, where CARRY8 blocks do not have a monotonic delay increase at their outputs. Our results demonstrate that security concerns raised by multitenant FPGAs are indeed valid and that countermeasures should be put in place to mitigate them.**

***Index Terms*—FPGA Security, Multitenancy, Power Analysis Attacks**

I. Introduction

Reconfigurable devices have become an appealing platform for a variety of applications, from small embedded and cyber-physical systems to large servers and data centers. In particular, the combination of speed, parallelism, flexibility, and energy efficiency offered by field-programmable gate arrays (FPGAs) has driven their adoption in the cloud backbone. Today, the largest cloud providers, such as Amazon (EC2 F1 [1]) and Microsoft (Catapult Project [2]), have already included FPGAs in their large-scale data centers.

These cloud systems are often multitenant: the available computational resources are shared among multiple system users. This scenario brings several challenges and raises concerns related, in particular, to security. In fact, it is necessary to guarantee the correct insulation between different users and it is fundamental to ensure that one user cannot interfere or delay the computation of another user. The above is not sufficient to guarantee security because it is also of utmost importance that information is not leaked through a side channel. However, this problem was limited to timing side channels, since the adversary would have needed physical access to the device to successfully mount all other types of physical attacks.

Recent research results demonstrated that this is no longer true, at least for power side-channel attacks [3], [4]. A power analysis attacker collects a number of power traces, typically with an oscilloscope connected to the power supply, guesses the secret key, and verifies the correctness of this guess over the collected power traces with statistical tools [5]. Voltage drops occurring during the computation of a cryptographic algorithm can also be captured by dedicated circuits realized in the FPGA logic itself, such as ring oscillators and delay lines. Data collected with these *sensors* could then be used instead of the classical power traces collected by an oscilloscope to successfully recover the secret key.

Successful examples of these attacks exploited directly the relation between circuit activity and the voltage fluctuations on the power distribution network (PDN) to extract encryption keys of a neighbouring encryption core [3], [4] or exploited the crosstalk coupling to extract values of secret data from neighbouring long wires [6], [7]. These attacks were carried out in a controlled lab environment with a dedicated setup and, while giving the intuition that the threat could also be exploitable in a real cloud environment, they were not providing any clear evidence of this fact.

In this paper, we demonstrate, for the first time, that a successful remote key recovery attack can be carried out on a real cloud system. We demonstrate this by attacking a 128-bit Advanced Encryption Standard (AES) cryptographic accelerator that we instantiated on an Amazon EC2 F1 instance. To collect the power traces, we designed a sensor based on a delay-line time-to-digital converter and adapted it to be tolerant to look-ahead carry chains (used in the Xilinx Virtex Ultrascale+ FPGA architecture of the Amazon EC2 F1 instances), which, unlike ripple-carry carry chains, do not have monotonic delay increase at their outputs. Our results, showing a successful key recovery attack on a real cloud system, demonstrate that the power side-channel vulnerabilities exist even in cloud FPGAs and that cryptographic circuits should not be left unprotected in a future multitenant scenario.

II. Remote Power Analysis Attacks on FPGAs

A. Threat Model

In a co-tenancy or multitenancy scenario, multiple users share the same reconfigurable logic and deploy their hardware tasks on it. Every deployed task and its computation is a relevant intellectual property of a user and, as such, must be

978-3-9819263-4-7/DATE20/©2020 EDAA

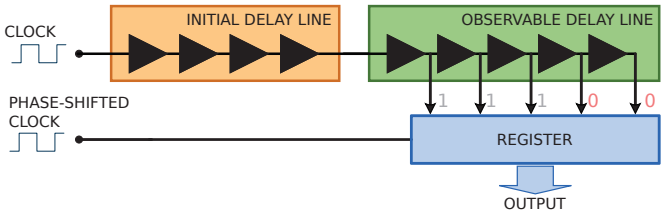

Fig. 1. Block architecture of a delay-line based voltage drop sensor.

protected. This is achieved with logical and physical separation of the tenants [8]. However, unwanted interaction between the tenants can still happen through the PDN. This can introduce several security risks, including the leakage of secret information via the power side channel.

In our paper, we assume that a victim uses a subset of the FPGA resources to encrypt data with a secret key, while the other tenant is malicious, with the aim of attacking the *confidentiality* of the victim. Logical and physical separation of the tenants ensure that the adversary is not able to connect to any of the signals in the victim logic—he has only the indirect access to the shared PDN. Furthermore, we assume that, after performing the encryption, the victim sends the ciphertexts over a public channel that can be observed by the adversary.

B. Power Analysis Attacks

Power analysis attacks have been introduced by Kocher et al. [5]. These attacks exploit the fact that power consumed by an integrated circuit depends on the switching activity of the logic cells. When the computation carried out is a cryptographic routine, an attacker can infer the secret key by exploiting this dependency. Although extremely powerful, power side-channel attacks have been always considered a danger only for embedded devices, since the adversary was required to have physical access to the device to collect the power traces needed to mount the attack. Schellenberg et al. [3] and Ramesh et. al [6] showed that this assumption no longer holds in some scenarios, including, at least potentially, the one of multitenant FPGAs.

C. Cloud FPGA-based Voltage Fluctuation Sensors

In case of FPGA development boards, it was shown that it is possible to measure internal voltage fluctuations on the PDN of an FPGA by using sensors realized in the reconfigurable fabric. Even though FPGAs contain system monitors, these on-chip measurement circuits have a limited sample rate and cannot detect nanosecond voltage fluctuations caused by logic switching at high-frequency [9], while sensors developed to remotely capture voltage fluctuations should be able to reach a resolution in the nanosecond range [9]. Side-channel voltage fluctuations measured this way provide means to mount a successful attack on a cryptographic accelerator running on the very same FPGA.

Two types of voltage drop sensors can be used to measure the power consumption in this way: the delay-line based sensors and the ring-oscillator based sensors. While both sensors have been used to attack cryptographic circuits on FPGAs [3], [4], ring-oscillator based sensors have a disadvantage that they are using combinational loops to measure the delay. In many scenarios, such as on the Amazon EC2 F1 instances studied in this paper, combinational loops are not supported [10]. This can be bypassed by designing ring-oscillator based sensors that contain registers, as it was shown by Giechaskiel et al. [11]. Despite this workaround, the sampling frequency of ring-oscillator based sensors is much lower than that of the delay-line based sensors, making the former unsuitable for recording nanosecond-scale voltage fluctuations needed to perform power analysis attacks. Therefore, our system uses delay-line based voltage drop sensors, described in the next paragraphs.

These sensors rely on the fact that all changes in the delay of CMOS logic gates are directly linked to the PDN voltage behavior, which thus indirectly exposes the switching activity and power consumption of the device. They have been presented for the first time by Zick et al. [9]. The simplest configuration consists of a line of buffers driven by a clock signal, as seen in Fig. 1. The propagation depth of the clock signal is proportional to the overall delay of the buffer line. The first part of the delay line, the *initial delay*, uses elements that have higher delay and less area overhead (such as LUTs or latches) to introduce a larger initial delay to the clock signal. To minimize the area overhead, only the last bits of the buffer chain are *tapped*, i.e, connected to registers that can record the state of the line. The registers are clocked with the same but shifted (delayed) clock signal, and therefore record how far the clock signal propagated through the delay line by the time the delayed clock reached the registers. The tapped part of the delay line, i.e, the *observable delay*, is usually implemented using carry-chain primitives that provide a very fine resolution per bit.

The sensor has to be calibrated so that the clock signal reaches the observable delay line before its state is saved to the registers. Similarly, to avoid saturation of the sensor output, the input clock should not reach the end of the observable line before its state is saved to the registers. This calibration includes changing the length of the initial delay or tuning the phase of the phase-shifted clock.

III. System Architecture and Experimental Setup

The target system of the attack that we present in this paper is an AES cryptographic core running on an Amazon EC2 F1 instance. Fig. 3 shows the simplified architecture of an Amazon EC2 F1 instance; it consists of a virtual machine (VM) with CENTOS operating system, having all the drivers and APIs needed for the communication with the FPGA. This VM can run any user code in C, including a correlation power analysis attack. The instance also contains a Xilinx Virtex Ultrascale+ FPGA that has a static *privileged shell*, used for controlling the communication between the user design and the C code in the VM.

Fig. 2 illustrates our system architecture, composed of several modules: first, an open-source 128-bit AES core [12]

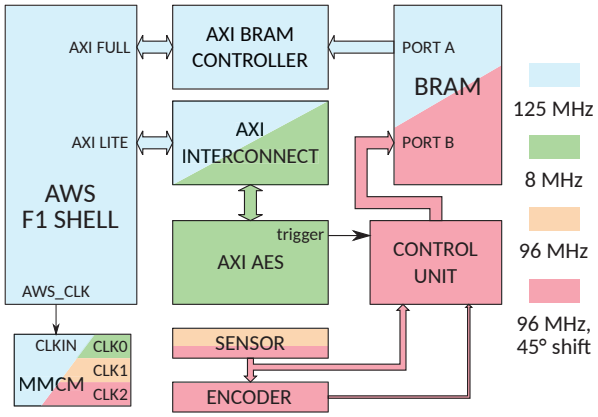

Fig. 2. System architecture.

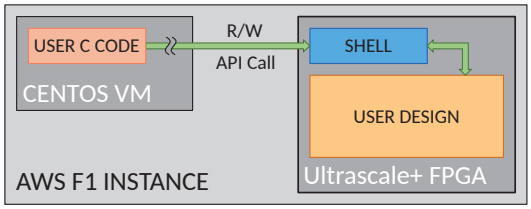

Fig. 3. Amazon EC2 F1 Instance architecture.

with an AXI-Lite wrapper. Then, a 160-bit voltage drop sensor to measure the power side-channel leakage, a priority encoder to encode the propagation depth of the clock to a corresponding binary number, and a true dual-port BRAM memory to store the sensor output. The remaining modules enable the communication between the AES, the BRAM, and the shell. Our system has four clock domains. The main clock (125 MHz) is provided by the shell; it drives the interconnect, the BRAM controller, and the read port of the BRAM. From this clock, all the remaining design clocks are derived. A 96 MHz clock drives the sensor delay line. A shifted 96 MHz clock is used for capturing the sensor output, the encoder, the control unit, and the BRAM write port.

The system simulates a potential victim by sending plaintexts to the AES core from the VM. Upon the start of each encryption, the AES core asserts a trigger signal and a number of sensor readings are encoded and stored in the BRAM. After the encryption, the ciphertext and the power consumption trace are communicated to the VM in two read requests. These steps are repeated for every new plaintext.

The implementation of our system satisfies the principle of physical separation, as the AES core and the sensor are separated by a column of unused DSP blocks. The use of the trigger signal from the AES could be avoided using trace alignment techniques on the sensor traces [3], in which case, the system would also satisfy the principle of logical separation.

The main challenge that we discovered while implementing the sensor on the Amazon EC2 F1 instance is that the delays of the CARRY8 outputs do not increase monotonically. In

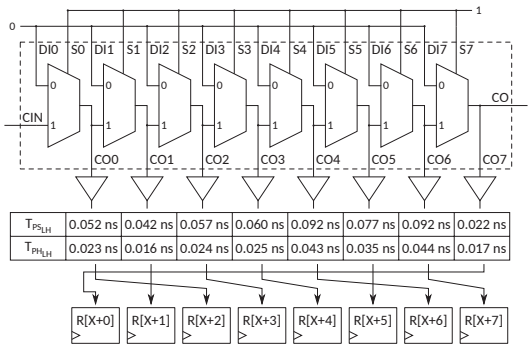

Fig. 4. CARRY8 internal architecture and nonmonotonic delays from input CIN to the outputs CO_i, $0 \leq i \leq 7$. To increase the likelihood of monotonically increasing delays with the increase in the CARRY8 output index, it suffices to permute the CARRY8 outputs before they reach the sensor register. Permutation is devised from a detailed timing analysis of all paths within one CARRY8 element.

other words, the delay from the input of the carry chain CIN to the individual carry outputs $CO_i, 0 \leq i \leq 7$, when all the other inputs are constant and set so that CIN can propagate, does not increase monotonically. In practice, this means that CO_7 output of the last reached CARRY8 in the observable delay-line can be high while its other outputs are low, producing unexpected sensor readings. After performing a detailed setup and hold timing analysis in Vivado, results of which are shown in Fig. 4, we conclude that, to achieve a monotonic increase in the output delays, CARRY8 outputs should better be arranged in the following order: CO_7, CO_1, CO_0, CO_2, CO_3, CO_5, CO_4, CO_6. To implement this, it suffices to permute the order of signals between the sensor register and the encoder, as illustrated in Fig. 4. Given that the system in Fig. 2 records the state of the entire observable delay line, we perform the permutation and the encoding in software and refer to the resulting traces as *permuted*. Conversely, we refer to the traces obtained using the encoded delay line values without permutations as *nonpermuted*.

IV. Experimental Results

The first experiment carried out was the analysis of the permuted power traces. Across all the samples in the collected traces, only five distinct values appeared: 113, 115, 116, 117, and 121, whereas the sensor operating range was 1–159. The waveform in Fig. 5 is obtained after averaging a hundred power-consumption traces collected from the sensor, where each trace corresponds to one encryption. Clearly, the plaintext loading (the first dip) and the subsequent ten rounds of AES encryption are all visible. Hence, the traces, albeit represented with only five distinct values, contain information that could be exploited by an attacker.

The second experiment was the correlation power analysis (CPA) attack on the permuted traces and each byte of the key during the last round of the AES encryption. The attack was mounted using 5×10^5 traces. Each byte attack lead to the successful recovery of the secret key byte. Fig. 6 illustrates the results of the attack on the third byte of the key. On the left

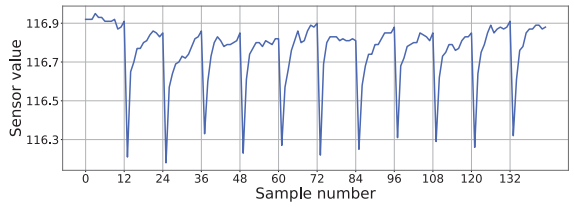

Fig. 5. Waveform obtained by averaging a hundred power-consumption traces. Plaintext loading and all ten AES encryption rounds can be clearly identified.

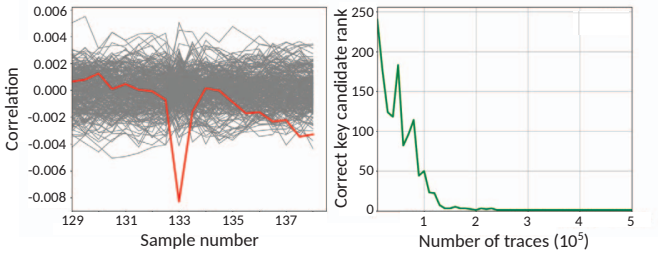

Fig. 6. CPA attack on the third byte of the AES encryption, using 5×10^5 permuted traces. On the left, the correlation for all key guesses. On the right, key rank evolution.

side, the correlation for the correct key guess (in red) reaches the maximum at sample 133 of the power trace. On the right side, we show how the ranking of the correct key guess evolves with the increase in the number of traces used in the attack. To guess all 16 key bytes, 5×10^5 traces were sufficient.

As the third experiment, we repeated the attack but this time on nonpermuted traces. Fig. 7 shows that the attack is still successful, although, the key rank drops to one after a much higher number of traces. In the case without permutation, 10^6 traces were required to guess the complete 16-byte key.

Finally, we repeated the previous two experiments 30 times. Before every experiment, we would shut down and restart the Amazon EC2 F1 instance, potentially having the resource allocator assign a different FPGA every time. The entire key was successfully broken in 42% of the attempts. Out of all attempts to attack the individual key bytes using 10^6 traces, 48% were successful. In 73% of the successful key-byte attacks, the CPA on the permuted traces succeeded with the smaller or equal number of traces than the CPA on the nonpermuted ones. Moreover, the CPA on the permuted traces required up to 88% (on average 20%) less measurements than the CPA on the nonpermuted traces, to retrieve one byte of the key. These variations were expected, as device timing characteristics are known to be affected by the process or temperature variations [13].

V. Conclusions

When deployed, multitenant FPGAs will offer new opportunities, but will also be exposing new security threats. Power analysis attacks are potentially among them, since recent work showed the possibility to remotely extract the secret key of a cryptographic algorithm by mounting an attack on the power traces obtained from sensors implemented using the FPGA

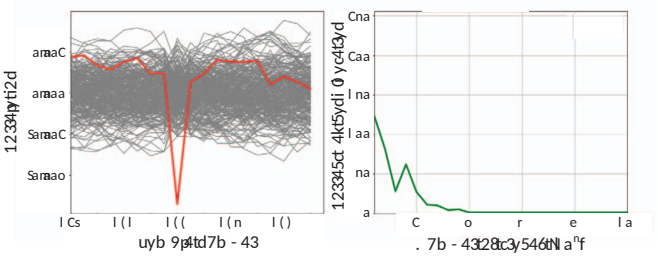

Fig. 7. CPA attack on the third byte of the AES encryption, using 10^6 nonpermuted traces. On the left, the correlation for all key guesses. On the right, key rank evolution.

fabric. These experiments were, however, mostly carried out on FPGA development boards in a controlled environment and not on a real system deployed on the cloud. In this paper, we demonstrate, for the first time, a successful key recovery attack on a cryptographic accelerator running on an Amazon EC2 F1 instance. As a case study, we used the AES-128 algorithm. However, our attack is applicable to any cryptographic core susceptible to power analysis attacks. Our results demonstrate that the security concerns raised by multitenant FPGAs are indeed valid and that countermeasures should be put in place to mitigate them.

References

[1] *Amazon EC2 F1*, Amazon AWS, 2019. [Online]. Available: https://aws.amazon.com/ec2/instance-types/f1/

[2] *Project Catapult*, Microsoft Research, 2019. [Online]. Available: https://www.microsoft.com/en-us/research/project/project-catapult/

[3] F. Schellenberg, D. R. E. Gnad, A. Moradi, and M. B. Tahoori, "An Inside Job: Remote Power Analysis Attacks on FPGAs," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, 2018, pp. 1111–1116.

[4] M. Zhao and G. E. Suh, "FPGA-Based Remote Power Side-Channel Attacks," in *Proc. of IEEE Symposium on Security and Privacy*, 2018, pp. 805–820.

[5] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO '99*. Springer, 1999, pp. 387–397.

[6] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb, and R. Tessier, "FPGA Side Channel Attacks without Physical Access," in *Proceedings of the 26th IEEE Symposium on Field-Programmable Custom Computing Machines*, 2018, pp. 45–52.

[7] I. Giechaskiel, K. B. Rasmussen, and K. Eguro, "Leaky Wires: Information Leakage and Covert Communication Between FPGA Long Wires," in *Proceedings of 13th ACM ASIA Conference on Information, Computer and Communications Security (ASIACCS)*, 2018, pp. 15–27.

[8] S. Trimberger and S. McNeil, "Security of FPGAs in data centers," in *IEEE 2nd International Verification and Security Workshop (IVSW)*, 2017, pp. 117–122.

[9] K. M. Zick, M. Srivastav, W. Zhang, and M. French, "Sensing Nanosecond-scale Voltage Attacks and Natural Transients in FPGAs," *Proceedings of the 21th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp. 101–104, 2013.

[10] *AWS EC2 FPGA GitHub*, Amazon, 2019. [Online]. Available: https://github.com/aws/aws-fpga/blob/master/ERRATA.md

[11] I. Giechaskiel, K. Rasmussen, and J. Szefer, "Measuring Long Wire Leakage with Ring Oscillators in Cloud FPGAs," in *Proceedings of the 29th International Conference on Field-Programmable Logic and Applications*, 2019, pp. 45–50.

[12] *AES Encryption Core*, AIST and Tohoku University, 2019. [Online]. Available: http://www.aoki.ecei.tohoku.ac.jp/crypto/

[13] D. R. Gnad, F. Oboril, S. Kiamehr, and M. B. Tahoori, "Analysis of transient voltage fluctuations in FPGAs," in *Proceedings of the IEEE International Conference on Field Programmable Technology*, 2016, pp. 1–8.