Securing FPGA Accelerators at the Electrical Level for Multi-tenant Platforms

Securing FPGA Accelerators at the Electrical Level for Multi-tenant Platforms

Tuan Minh La, Kaspar Matas, Khoa Dang Pham, and Dirk Koch
The University of Manchester, UK
{tuan.la, kaspar.matas, khoa.pham, dirk.koch}@manchester.ac.uk

***Abstract*—As FPGAs are now offered on the cloud, this exposes many potential security issues. This PhD project investigates current security issues and challenges when deploying FPGAs in the cloud as well as using FPGAs in a multi-tenancy scenario. By addressing practical threats, and most importantly, proposing feasible countermeasures, this paper shows preliminary results on protecting FPGAs for multi-tenant scenarios.**

***Index Terms*—FPGA, reconfigurable computing, FPGA hardware security**

I. Introduction

FPGAs are widely offered in cloud data centers (e.g., [1]), and there is consequently a strong need to investigate FPGA hardware security. As in a cloud service scenario, anybody can access an FPGA. This may enable an attacker to cause potential harm to the FPGA equipment [2] (see Figure 1).

In order to prevent potential malicious designs being deployed on an FPGA instance, cloud service providers commonly demand uploading netlists in order to run design rule checks (DRCs). Tests include static timing analysis and detecting *Combinatorial feedback paths*. These violations are currently used for FPGA attacks [2]. For example, *DRC LUTLP** in the Xilinx tools indicates a LUT-based combinational loop, and if not mitigated, this may create race conditions or free-running oscillators which can be used to leak information or environmental parameters of a system. Additionally, accepting netlists only ensures that generated bitstreams are not corrupted through any post-processing step. As a last defense, power monitoring can be used to warn or stop potential malicious activities (e.g., through clock gating if the supply power consumption reaches a critical level [3]).

DRCs provided by the FPGA vendors are insufficient for security, as shown with several examples of malicious FPGA designs that can be deployed on AWS instances in Figure 2. However, monitoring is passive as it only detects malicious circuits *after it is deployed*.

The next section will survey current security issues of using FPGAs in the cloud and solutions to mitigate those. This work not only aims to provide the base for future multi-tenancy scenarios but also to move FPGA usage from the present FPGA-enabled Acceleration-as-a-Service (AaaS) offerings [1] to full FPGA-as-a-Service (FaaS) offerings [4], [5], where users may upload own bitstream.

II. Cloud FPGA Security Threats

Cloud computing is not an uncommon target for abuse of security flaws. Amazon Elastic Cloud Computing (EC2) has a record of security (e.g., [6]). Moreover, hardware flaws threatening cloud infrastructures include rowhammer [7] attacks on

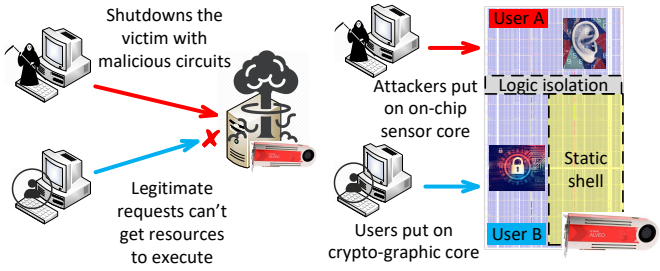

Fig. 1. Attacks on system availability and system confidentiality.

DDR memory or Spectre [8] and Meltdown [9] which are exploiting side channels hidden through speculative execution.

The recent trend of cloud service providers to offer FPGAs has stimulated significant research in hardware security. Interestingly, virtually all recently published attacks on FPGAs use oscillators in some way. Oscillators help to precisely measure system states [10] as well as a circuit drawing excessive waste power (also known as power-hammering) [11].

Through wasting power, some malicious designs cause voltage drops through high-frequency switching activity. One of the easiest ways to generate high switching activity is through ring oscillators. A ring oscillator usually consists of an odd number of NOT gates. Our measurements in [12] show frequencies of ring-oscillators close to 6GHz, which is much faster than any real-world FPGA design would ever toggle. This is a primary reason why Amazon prohibits combinatorial loops in user designs [13]. However, there have been papers published showing ring oscillator designs that bypass the Xilinx vendor Design Rule Checking (DRC) [14], and the ring oscillators shown in Figure 2 are currently deployable on Amazon. They use 1) transparent latches, 2) asynchronous sets/resets, 3) FPGA primitives not considered for detecting combinatorial feedback loops (e.g., carry-chain logic, cascading multiplexers in Xilinx FPGA slices (F7MUX or F8MUX), or DSP blocks) [12], or 4) glitch amplification effects.

Glitches can be amplified through XOR gates [15], which has the property that any change at the input causes a change at the output. Therefore by adjusting routing delays, it is possible to physically implement an oscillator where the same source of a toggle flip-flop is routed to an XOR with different delays to create glitches that, in turn, are fed back to the toggle flip-flop for creating self-oscillation (Figure 2d).

We experimentally deployed all these oscillators on AWS F1 instances and can confirm that they can bypass DRCs and run as expected. AWS has a protection mechanism built in their shell (the static system of their instance that never changes) that gates the system clocks whenever a user exceeds their

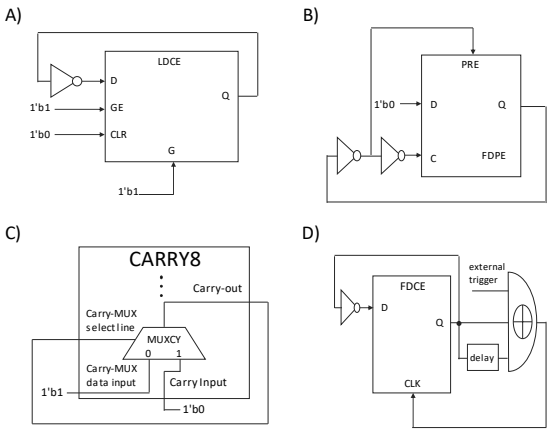

Fig. 2. Oscillator designs deployable on AWS F1 instances: (a) transparent latch, (b) flip-flop with asynchronous reset, (c) carry-chain ring oscillators, (d) a self-oscillating circuit using glitch amplification.

power or temperature envelope. However, by replacing the system clock by a ring-oscillator, we had been able to bypass this protection mechanism.

III. Proposed Mitigation Mechanism

Vendors suggest monitoring the FPGA in order to detect voltage drops and high current. However, this approach has the disadvantage that an attack is flagged *after* it happened. In order to prevent power-hammering and other kinds of attacks in the first place, it is better to enhance the vendor DRC checks to detect malicious designs *before* deployment. A first work proposing this approach [16], showed that ring oscillators could be detected directly in bitstreams for Lattice FPGAs. In [12] and [17], we introduced the open-source FPGA virus-scanner framework FPGADefender which allows checking for malicious constructs at the bitstream level [18] (see Figure 3) for all UltraScale+ FPGAs from the vendor Xilinx. With this in place, it would not be necessary to upload netlists to Amazon AWS (and with this exposing IP). Currently, FPGADefender virus scanner checks include:

- *Combinatorial Feedback Loop Detector.*
- *Fanout Detection.*
- *Disallowed Port Detection.*
- *Disallowed Path Detection.*
- *Short Circuit Detection.*

These detectors generate numerical scores that describe the maliciousness of a design, with a low score indicating a relatively benign design and a high score meaning the design employs many problematic components. These scores are then combined to produce a final score, which can then be used by the runtime system to decide if the design should be loaded.

IV. Conclusions

In conclusion, this PhD project continues exploring security issues on FPGAs. Alternative oscillators have been reported recently, but cloud service providers and vendor companies have not made any updates to mitigate these threats. This allows various remote attacks at the electrical level, which eventually can crash (or damage) the FPGA or the entire instance. Like DRC checks, our FPGADefender scanner is a proactive approach to protect an FPGA before any bitstream loading process.

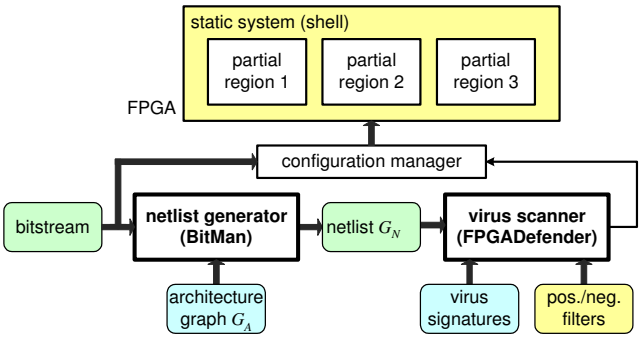

Fig. 3. The FPGADefender Framework. The input is a bitstream (alternatively a netlist) that is analyzed by a virus scanner engine, parameterized through virus signature files. The test result allows a configuration manager deciding if a configuration is allowed to be loaded onto the FPGA or not.

With this work, we want to contribute to the currently very active research on FPGA hardware security in order to understand threat scenarios better and for providing mitigation mechanisms. While there is more research needed, we firmly believe that multi-tenancy as well as FPGA-as-a-Service can be accepted from a security point of view, and that bitstream scanning is the key technology to allow this.

Acknowledgement

This work is kindly supported by the UK National Cyber Security Centre through the project *rFAS* (grant agreement 4212204/RFA 15971). We also thank the Xilinx University Program for providing tools and boards.

References

[1] Amazon EC2 F1 Instances. [Online]. Available: https://aws.amazon.com/ec2/instance-types/f1/
[2] S. S. Mirzargar et al., "Physical Side-Channel Attacks and Covert Communication on FPGAs: A Survey," in *FPL'19*.
[3] AFI Power. [Online]. Available: https://github.com/aws/aws-fpga/blob/master/hdk/docs/afi_power.md
[4] R. Watanabe et al., "Implementation of FPGA Building Platform As a Cloud Service," in *HEART'19*.
[5] Deep Dive into Alibaba Cloud F3 FPGA as a Service Instances. [Online]. Available: https://www.alibabacloud.com/blog/deep-dive-into-alibaba-cloud-f3-fpga-as-a-service-instances_594057
[6] N. Gruschka et al., "Vulnerable cloud: Soap message security validation revisited," in *2009 IEEE International Conference on Web Services*.
[7] Y. Kim et al., "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," in *ISCA'14*.
[8] P. Kocher et al., "Spectre Attacks: Exploiting Speculative Execution," in *S&P'19*.
[9] Moritz Lipp et al., "Meltdown: Reading Kernel Memory from User Space," in *USENIX Security 18*.
[10] K. Zick et al., "Low-cost Sensing with Ring Oscillator Arrays for Healthier Reconfigurable Systems," *ACM TRETS*, August 2012.
[11] D. Gnad et al., "Voltage Drop-based Fault Attacks on FPGAs Using Valid Bitstreams," in *FPL'17*.
[12] K. Matas et al., "Invited Tutorial: FPGA Hardware Security for Datacenters and Beyond," in *FPGA'20*.
[13] Aws ec2 fpga hdk+sdk errata. [Online]. Available: https://github.com/aws/aws-fpga/blob/master/ERRATA.md
[14] I. Giechaskiel et al., "Measuring Long Wire Leakage with Ring Oscillators in Cloud FPGAs," in *FPL'19*.
[15] K. Matas et al., "Power-hammering through Glitch Amplification - Attacks and Mitigation," in *FCCM2020*.
[16] J. Krautter et al., "Mitigating electrical-level attacks towards secure multi-tenant fpgas in the cloud," *TRETS*.
[17] T. M. La et al., "FPGADefender: Malicious Self-Oscillator Scanning for Xilinx UltraScale+ FPGAs," *ACM TRETS*, 2020.
[18] K. Pham et al., "BITMAN: A Tool and API for FPGA Bitstream Manipulations," in *DATE'17*.