Reading Between the Dies: Cross-SLR Covert Channels on Multi-Tenant Cloud FPGAs

Ilias Giechaskiel
University of Oxford
Oxford, United Kingdom
ilias.giechaskiel@cs.ox.ac.uk

Kasper Rasmussen
University of Oxford
Oxford, United Kingdom
kasper.rasmussen@cs.ox.ac.uk

Jakub Szefer
Yale University
New Haven, CT, USA
jakub.szefer@yale.edu

***Abstract*—Field-Programmable Gate Arrays (FPGAs) are becoming increasingly available via commercial cloud providers, which currently allocate devices on a per-user basis. As the underlying hardware is often underutilized, several proposals for multi-tenant use of FPGA resources have been brought forth, along with some initial work on security attacks in this setting. Simultaneously, high-end FPGAs are being produced with 2.5D integration of multiple distinct dies, called Super Logic Regions (SLRs), onto the same chip. Although one might expect that physical separation of logic onto separate dies could prevent multi-tenant attacks, this paper demonstrates for the first time that cross-SLR information leaks based on sensing voltage changes within the FPGA chip are possible, without physical access to or modification of the boards. The cross-SLR covert channel is characterized analytically and experimentally on five Xilinx Virtex UltraScale+ FPGAs, both locally and on the Amazon and Huawei clouds. Several configurations of the source transmitters and the sink receivers are tested, including their locations, types, and sizes. The power-based channel is shown to have a bandwidth upwards of 4.6 Mbps and accuracy of over 97.6%. Consequently, as physical separation of tenants onto separate dies (SLRs) is an insufficient countermeasure against information leaks, hardware-level architectural improvements are necessary to make secure multi-tenant FPGAs on shared clouds a reality.**

***Index Terms*—Cloud and virtualized FPGAs, multi-tenant FPGAs, ring oscillators, super logic regions, covert channels**

I. Introduction

Field-Programmable Gate Arrays (FPGAs) have grown tremendously in size over the last decade: $16\,\mathrm{nm}$ Virtex UltraScale+ devices from 2019 can contain over 4 million lookup-tables (LUTs) and 8 million flip-flops (FFs) [42], compared to less than 500 thousand LUTs and 1 million FFs in $40\,\mathrm{nm}$ Virtex 6 devices [44] from 2009. In addition, with more cloud providers offering FPGAs to end users, and with FPGA chips integrated with CPU processors, virtualized multi-tenant FPGAs are becoming necessary to make better use of the underlying physical resources [6], [7], [9], [21]–[23], [38], [39]. However, having co-located designs from multiple users on a single FPGA raises several security concerns in light of recent attacks without physical access to the FPGA board [11]–[14], [24], [25], [27]–[31], [36], [47].

Although isolation is often proposed as a step to mitigate potential information leakage [11]–[13], [28], [37], several attacks have been successful, despite physical separation of the adversarial and victim logic [24], [25], [30], [47]. One limitation of these attacks is that they target low-end FPGA devices, where physical isolation is not strong: the transmitting and receiving circuits share the same FPGA die, and, in some cases, even the same clock regions and resources [25], [47].

However, more advanced FPGAs (namely those in the Xilinx Kintex UltraScale and Virtex 7, UltraScale, and UltraScale+ families) are now available, containing multiple dies incorporated into the same FPGA chip. These distinct dies, called Super Logic Regions (SLRs), could be used to multiplex the FPGA on a per-SLR basis among different cloud FPGA tenants. Although this form of physical isolation per tenant may appear to be stronger and a potential security improvement, in this paper we show that it is not sufficient to prevent information leaks across the SLR dies. Specifically, we introduce the first successful cross-SLR attack, which we demonstrate through a covert channel on a local board and on Amazon and Huawei FPGA servers. We characterize the resulting covert channel analytically and experimentally, and show that it has a bandwidth of $4.6\,\mathrm{Mbps}$, with over 97.6% accuracy. It also remains fast and accurate across many types of ring oscillator transmitters and receivers. As current cloud providers allocate the FPGA on a per-user basis, our attack does not yet affect Amazon and Huawei users in practice. However, it demonstrates that before cloud providers can start implementing multi-tenant FPGAs, architectural improvements in the hardware layer are necessary.

II. Background: Stacked Silicon Interconnect

Xilinx FPGAs have been available on public clouds since 2016, when Amazon Web Services (AWS) announced Elastic Cloud Compute (EC2) F1 instances with Xilinx Virtex UltraScale+ FPGAs [3]. Other public cloud providers soon followed, with Virtex UltraScale+ FPGAs offered on Huawei FPGA-Accelerated Cloud Server (FACS) FP1 instances [45] and Alibaba Cloud F3 instances [1]. Moreover, Kintex UltraScale boards are available on Baidu FPGA Cloud Compute [4] and Tencent Cloud FX2 instances [35]. These high-end boards use a Stacked Silicon Interconnect (SSI) technology to create much larger devices with a lower power envelope and more dedicated features [41], [42]. The SSI allows multiple distinct dies (SLRs) to be integrated into one big FPGA chip.

The Amazon and Huawei FPGA clouds investigated in our experiments use $16\,\mathrm{nm}$ Virtex UltraScale+ XCVU9P chips, which split their 90 clock regions equally into three separate SLR dies. These SLRs are adjacent to each other, and are connected through a silicon interposer, as shown in Figure 1. This

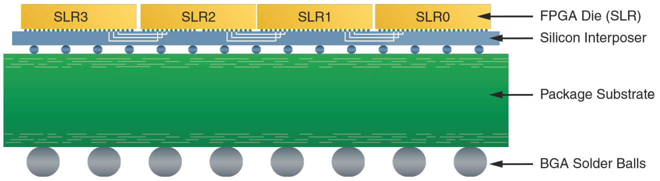


Fig. 1: Stack Silicon Interconnect (SSI), adapted from [41]. Super Logic Regions (SLRs) are separate FPGA dies, connected and powered through the silicon interposer, which acts as a conduit to external I/O through the package substrate.

interposer is a passive layer which connects global clocking and general interconnect resources to the SLR dies [41]. It also acts as a conduit between SLR components and the package substrate, providing connectivity to I/O pins, and power and ground connections using Through-Silicon Vias (TSVs) [41].

Each SSI device has a master SLR die, which "initiates configuration of the device and all other SLR components" [41]. The master SLR also has access to some dedicated FPGA circuitry, including the Analog-to-Digital Converter (XADC) and unique identifiers, such as the Device DNA and User eFUSE [41]. Cloud FPGAs making use of partial reconfiguration reserve portions of the master SLR die (and of slave SLRs) for their "shell" interface, which abstracts away concrete physical implementation details such as I/O pinouts, DRAM controllers, and clock logic.

This SLR layout could also provide a natural partitioning mechanism for multi-tenant cloud FPGAs, with the cloud provider reserving the master SLR, and different user designs being restricted to separate SLR dies. This would result in better physical isolation between the different users compared to existing proposals which split logic along (or even within) clock regions on the same die [21]–[23]. However, as we show in Section V, this physical isolation along SLR dies is still insufficient to prevent cross-SLR communication.

III. System and Adversary Model

With several proposals for virtualized FPGAs in cloud environments [6], [7], [9], [21]–[23], [38], [39], it is important to examine sources of potential data exfiltration (e.g., of cryptographic keys or other sensitive information) between users that share the same reconfigurable device. This paper is therefore concerned with intentional leakage for covert communication (as opposed to unintentional side-channel attacks) between FPGA logic that is logically and physically separated.

Prior work has shown that physical isolation within a single die can result in information leaks (Section VIII). These leaks are caused by voltage drops, with the strength of the effect (magnitude of the drop) decreasing with increasing distance between the transmitter and the receiver, at least for monolithic FPGAs [27]. As a result, even though the power rails between different SLRs may be shared, logic is placed much farther apart compared to intra-SLR circuits. It is thus reasonable to wonder whether communication between user logic placed on different SLR dies, as shown in Figure 2, remains possible.

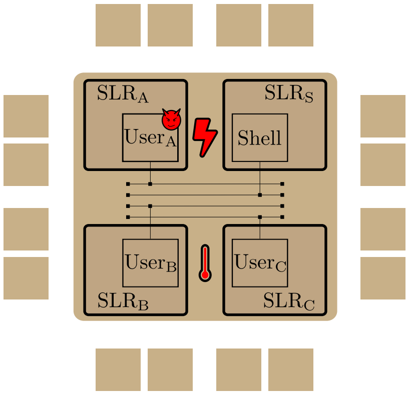


Fig. 2: System model for multi-tenant FPGAs. Malicious and benign user designs share the reconfigurable fabric, but are logically and physically isolated on separate SLRs. Adversarial logic can act as side-channel receivers and transmitters by influencing voltage and/or temperature.

As our target scenario is that of cloud FPGAs, adversaries do not have physical access to the hardware, and are restricted to using well-defined interfaces, such as those provided by a cloud shell. Consequently, adversaries cannot directly read temperature and voltage conditions (provided by system monitors), but may attempt to infer or influence them indirectly. As cloud providers allow attackers to place and route logic within the confines of their dedicated regions, custom placement and routing is also allowed by our threat model. However, as explained in Section VII-A, this is not crucial for the success of our covert channel. Attacks on the FPGA software tools and the bitstream itself are out-of-scope.

We primarily investigate how to exploit information leakage for covert communication, and focus on multi-tenant cloud FPGAs, which have attracted the interest of the security community [10]–[13], [28], [29], [47]. However, our system model is equally applicable to covert communication between potentially outsourced untrusted third-party Intellectual Property (IP) cores [11], [12], [17], [18], and System-on-Chip (SoC) FPGAs [5], [11], [12], [32], [47], for instance those found in Intel Xeon CPUs with integrated FPGAs [20], Xilinx Zynq UltraScale+ MPSoC FPGAs with hard ARM processors [42], or Microsemi FPGAs with soft RISC-V processors [26].

IV. Experimental Setup

In this work, we test the hypothesis that activating large circuits results in a voltage drop that is measurable across SLR dies. We use ring oscillators (ROs) as transmitters, and vary their frequency by changing the number of RO stages. We also use ROs as receivers, since they are sensitive to process, voltage, and temperature (PVT) variations [16]. We demonstrate the generality of our covert channel by testing SLR locations and RO types on a local VCU118 evaluation board, and on two FPGAs on each of the Amazon AWS and

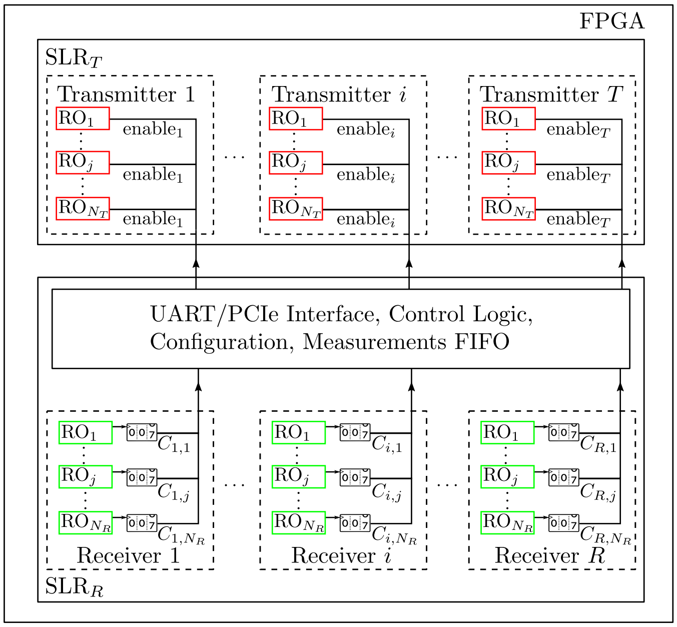


Fig. 3: Block diagram of the experimental setup, illustrating the logical and physical isolation of the $R \cdot N_R$ RO receivers and the $T \cdot N_T$ RO transmitters on separate SLRs.

Parameter	**Local**	**AWS F1**	**Huawei FP1**
FPGAs Tested	1	2	2
Board	VCU118	Proprietary	Proprietary
XCVU9P Chip	flga2104-2-e	flgb2104-2-i	flgb2104-2-i
Shell Clock Regions	None	X4Y0:X5Y9	X3Y4:X5Y9
Comb. Loops Allowed	Yes	No	Yes
Clock Frequency (MHz)	300	125	200
Communication	UART	PCIe	PCIe
Vivado Version	2017.4	2018.3	2017.2

TABLE I: Hardware parameters of the FPGA boards.

Huawei clouds. All five boards contain Virtex UltraScale+ XCVU9P chips with three SLRs, with cloud shells reserving parts of SLR dies 0 and 1 (SLR 1 being the master SLR [43]).

In our setup, we employ T independent transmitters, each containing N_T ROs, and each placed on separate clock regions of the same SLR, SLR_T. On the receiving SLR, SLR_R, we instantiate R receivers on different clock regions, each of which contains N_R ROs, whose frequency we estimate by using counters. These counters and other control logic are placed on separate clock regions of $\texttt{SLR}_R$. Measurements are transferred to a PC over the UART for the VCU118 board and over PCIe for the two cloud providers. Figure 3 shows a diagram of the experimental setup.

In accordance with our threat model, we do not modify the boards, and use the default clock configuration of each device. We also do not control the placement and routing of any transmitter RO within their assigned clock regions, and only control the placement of receiver ROs to identify the effect of distance on the accuracy and bandwidth of the communication channel. We summarize the properties of our experimental setup in Table I, and show an example instantiation of the measurement architecture in the Appendix.

Parameter	**Value**
Receivers, R	5
ROs per Receiver, N_R	5
Transmitters, T	†12
ROs per Transmitter, N_T	500
Receiver SLR, SLR_R	1
Transmitter SLR, SLR_T	0
RO Type	LD
RO Buffer Stages	2
Measurement Cycles	2^7

TABLE II: Fixed (top) and variable (bottom) experimental parameters. † $T = 8$ for the local VCU118 board.

For all setups, we use $R = 5$ receivers with $N_R = 5$ ROs each, and $T = 12$ transmitters ($T = 8$ for the local board): as the cloud providers reserve some clock regions for their shell, the three setups are not identical, but without any impact on the quality of the communication channel (Section V). These fixed properties are shown in the top of Table II.

As AWS prohibits combinatorial loops from user designs [2], we use alternative RO designs introduced recently [13], [33]. Although traditional ROs use only combinational logic through lookup-tables (LUT-RO), by replacing a buffer stage with a latch (LD-RO) or a flip-flop (FF-RO), one can overcome restrictions placed by cloud providers today. These three types of ROs are shown in Figure 4. By default, we use LD-ROs with two additional intermediate buffer stages for the receiver and the transmitter ROs.

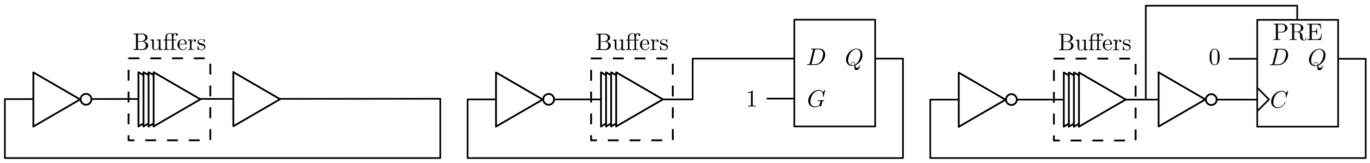
As shown in the bottom of Table II, we initially place the receivers on SLR 1, the transmitters on SLR 0, and instantiate $N_T = 500$ ROs per transmitter. Moreover, we count the number of RO signal transitions during a 2^t clock-cycle period with $t = 7$, corresponding to 0.4-1.0 µs, depending on clock speed. Sections V and VI vary these parameters. For each setup, we run five tests of $2,048$ measurements each, collecting $10,240$ data points from each RO per testing configuration.

V. Leakage Characterization

In this section, we introduce metrics to measure the information leakage (Section V-A), and characterize it across different transmitter sizes (Section V-B), SLR locations (Sections V-C), and types of transmitting and receiving ROs (Section V-D).

A. *Measurement Metrics*

To quantify the leakage, we compare the average RO count for a receiver R_i when the transmitters are disabled (C_i^0) and when they are enabled (C_i^1). We plot the difference $\Delta C_i = C_i^0 - C_i^1$ across all $R \cdot N_R = 25$ ROs per FPGA for the default setup (Table II) in Figure 5. Since $\Delta C_i > 0$ for all i, the receiver can easily distinguish between transmissions of 0 and 1, with fewer than $5/127,875 = 0.004\%$ misclassifications per FPGA board (the encoding scheme and classification algorithm are described in Section VI). Moreover, Figure 5 illustrates that for a given RO R_i, ΔC_i is close for identical boards, with small shifts accounting for process variations.



(a) LUT-RO: Lookup-table ring oscillator

(b) LD-RO: Latch ring oscillator

(c) FF-RO: Flip-flop ring oscillator

Fig. 4: Three ring oscillators designs with a variable number of intermediate buffer stages, used in local and cloud experiments. Although LUT-ROs (a) are prohibited on Amazon Web Services, LD-ROs (b) and FF-ROs (c) are not.

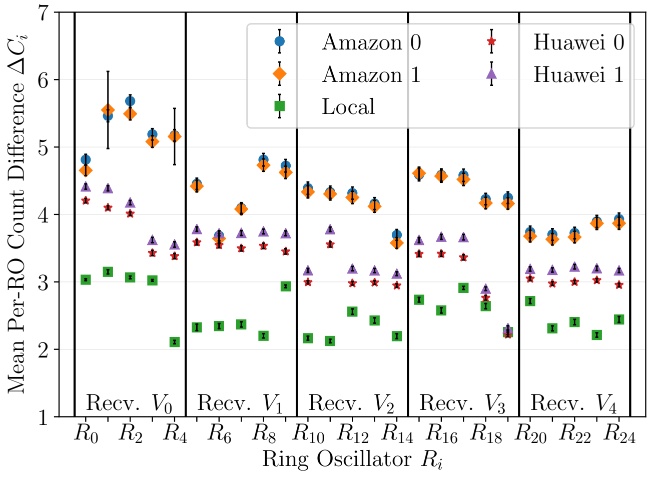


Fig. 5: Average count differences ΔC_i in the default setup across the 25 ROs per FPGA, with 99% confidence intervals.

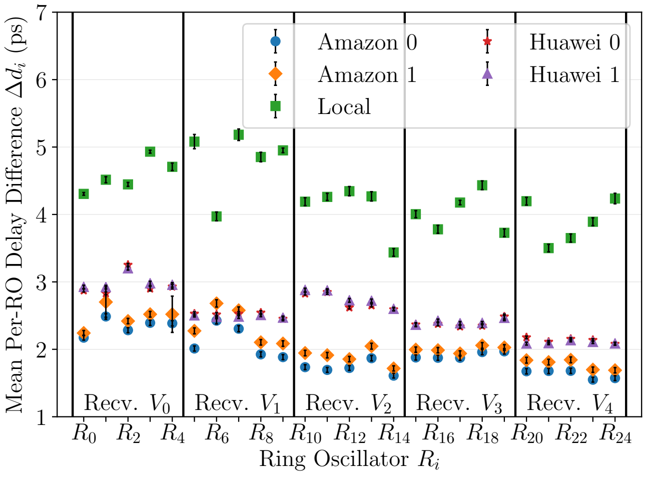


Fig. 6: Average delay differences Δd_i in the default setup across the 25 ROs per FPGA, with 99% confidence intervals.

As this metric is sensitive to the clock and RO frequencies, it does not allow for meaningful comparisons within or across FPGA boards. To overcome this limitation, we can estimate the absolute delay difference Δd_i of each RO by accounting for the clock frequency f_c and the measurement period 2^t, by adapting an equation derived by Giechaskiel et al. [13]:

$$\Delta d_i = \frac{2^t}{2 f_c} \cdot \frac{C_i^0 - C_i^1}{C_i^0 C_i^1} \tag{1}$$

Figure 6 plots the absolute delay differences Δd_i in picoseconds (ps) using Equation (1). For any board, there is less variation in the strength of the effect within a receiver V_j compared to the variation between receivers. Moreover, the average delay Δd^j of the five ROs in receiver V_j generally follows the distance of the receiver to the transmitters, i.e., $\Delta d^{\{0,1\}} > \Delta d^{\{2\}} > \Delta d^{\{3,4\}}$, with V_2 being an exception in the Huawei boards. Moreover, we note that boards with a faster clock frequency and a smaller shell are affected more. These effects might be attributed to increased switching activity and out-of-sync competing logic respectively. However, it is not possible to conclusively determine why the three FPGA setups behave differently, as the strength of the leakage also depends on the FPGA board voltage regulator.

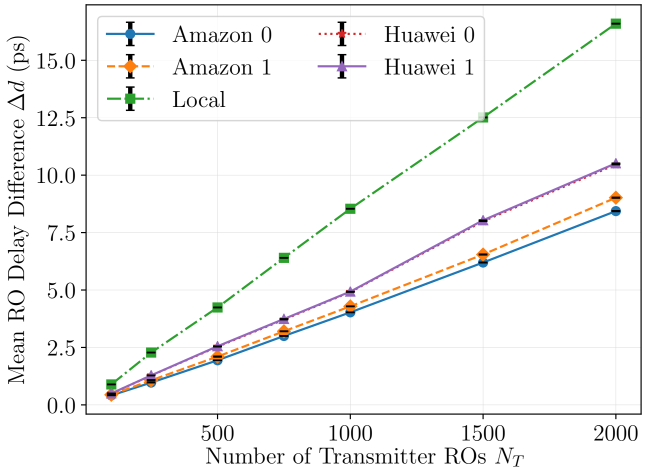


Fig. 7: Average delay differences Δd for different transmitter sizes N_T, with 99% confidence intervals.

B. Transmitter Sizes

To further understand the communication channel, we vary the size of the transmitting circuits. Figure 7 plots the delay difference Δd averaged over all 25 ROs for different numbers of transmitting ROs N_T and all five FPGAs. As expected,

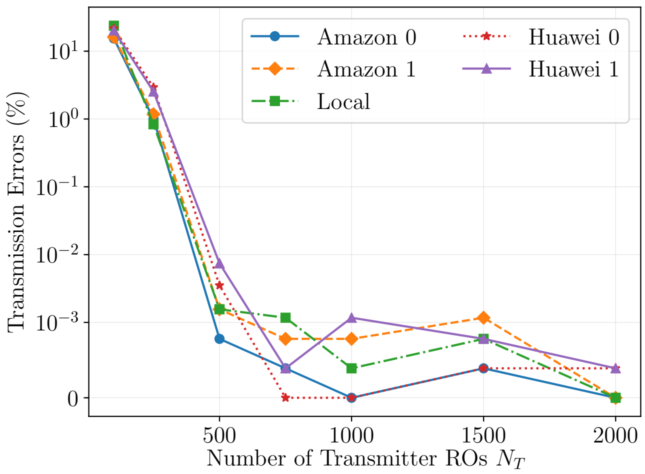


Fig. 8: Number of transmission errors for different sizes N_T.

more transmitting ROs result in larger voltage drops, and therefore larger changes in the RO frequency. Although the local VCU118 board was designed with fewer transmitters ($T = 8$ due to differences in the communication logic over the UART instead of $T = 12$ for the cloud FPGAs over PCIe), the effect is stronger, as also explained in Section V-A. It should be noted that although we have chosen to average over all ROs, other statistics can also be used. For example, the median, the sum, or even a fixed choice of RO, all result in similar graphs.

To understand how a larger Δd (and also the amount of transmitter logic/area) affects the channel accuracy, we plot the total number of misclassifications for various transmitter sizes in Figure 8. A transmitter size of $N_T = 100$ results in correct classifications over 75% of the time, while increasing the number of transmitters to $N_T = 250$ results in an accuracy of over 97%. Further increasing the transmitter size to $N_T \geq 500$ reaches accuracies of over 99.9%.

C. Transmitter and Receiver SLR Locations

To ensure that the covert channel is present on all locations on the FPGA, we vary the SLRs on which the receivers SLR_R and transmitters SLR_T are placed, and plot the results in Figure 9. For all SLR combinations, the leakage remains measurable, with $\Delta d > 0$. Figure 9 also confirms that when the transmitters and the receivers are two SLRs apart, Δd is smaller than when they are only one SLR apart. The other four placements result in similar Δd, except for $(SLR_R, SLR_T) = (2, 1)$ in the cloud, potentially due to the dynamic activity of the shell. Accuracy remains over 99.9% for all setups.

D. Ring Oscillator Properties

The next set of experiments investigates the effect of the receiver and transmitter RO types. Figure 10 shows the change in delay for all nine such type combinations (four on Amazon, as LUT-ROs are prohibited [2]). First of all, Figure 10 indicates that all three types of ROs are effective as both transmitters and receivers, since $\Delta d > 0$. Moreover, some consistent patterns emerge. For example, for a fixed receiver type, the FF-RO transmitter results in a larger Δd, as it has more stages. Similarly, for a fixed transmitter type, a receiver FF-RO is affected more than an LD-RO and a LUT-RO, as it has more stages which are influenced by the voltage drop. By contrast, the count difference ΔC follows the opposite pattern, as FF-ROs are the slowest, due to their extra inverter stage. Accuracy again remains over 99.9% for all setups.

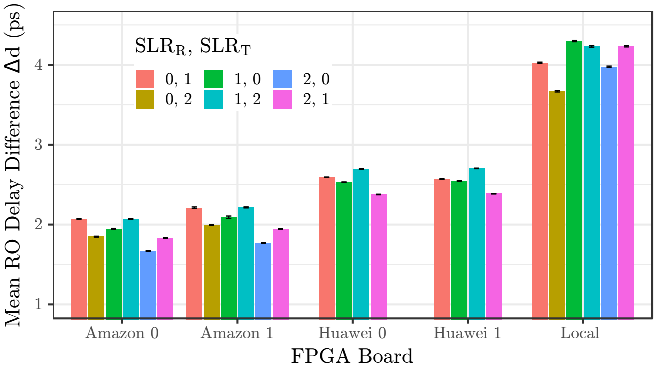


Fig. 9: Average delay differences Δd for different receiver (SLR_R) and transmitter (SLR_T) SLRs, with 99% confidence intervals. Placement and routing failed when the receiver and transmitter were two SLRs apart in the Huawei boards.

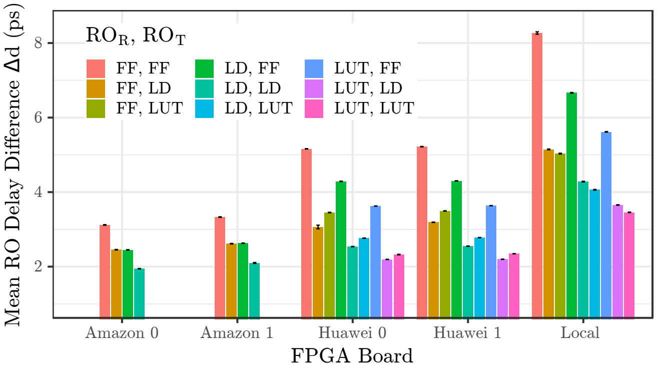


Fig. 10: Average delay differences Δd for different receiver (RO_R) and transmitter (RO_T) RO types, at 99% confidence.

The effect of additional stages is highlighted in Figures 11 and 12, which vary the number of intermediate buffers in the receiver and transmitter ROs respectively. Figure 11 shows that more receiver buffer stages result in higher Δd. However, because the RO frequency decreases, so does ΔC, resulting in more errors: with 9 stages, the error increases to over 2%, and exceeds 26% with 15 stages. Moreover, at 16 intermediate buffer stages, ROs can no longer fit in a single Configurable Logic Block (CLB), even when using dual output `LUT6_2` lookup primitives. Increasing the number of transmitter stages (Figure 12) generally increases Δd but at a decreasing rate. This is due to a tradeoff between the amount of logic activated on a transmission of 1 (which increases), and the switching frequency of the logic (which decreases). The errors for ≥ 1

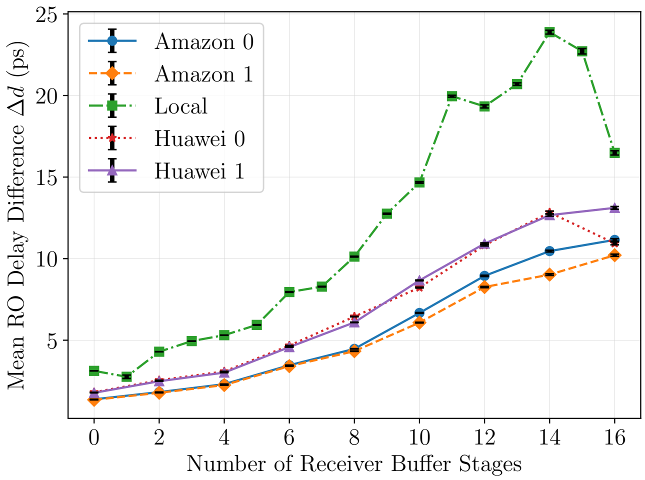


Fig. 11: Average delay differences Δd for different intermediate buffer stages in the receiver ROs, at 99% confidence.

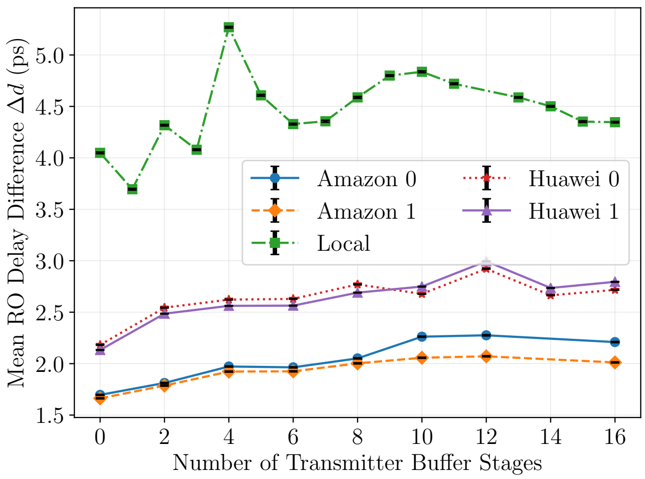


Fig. 12: Average delay differences Δd for different intermediate buffer stages in the transmitter ROs, at 99% confidence.

transmitter stages remain consistently below 0.01%.

VI. Bandwidth Analysis

Having characterized this strong source of cross-SLR information leakage, we now estimate the bandwidth of the ensuing covert channel. We first discuss the encoding scheme and resulting bandwidth in the basic use-case (Section VI-A), and then examine multi-bit transmissions (Section VI-B).

A. *Encoding Scheme*

In some setups, e.g., the VCU118 board, a simple threshold is sufficient to reach accuracies of almost 100%. However, these thresholds vary per RO, require calibration, and are sensitive to environmental conditions. This is shown in Figure 13, which plots raw RO counts from an AWS experiment.

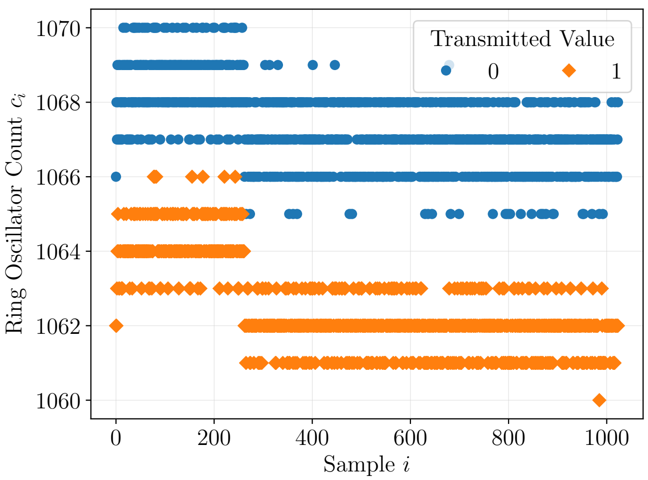


Fig. 13: Raw RO counts for an AWS experiment. A simple threshold cannot always account for environmental conditions.

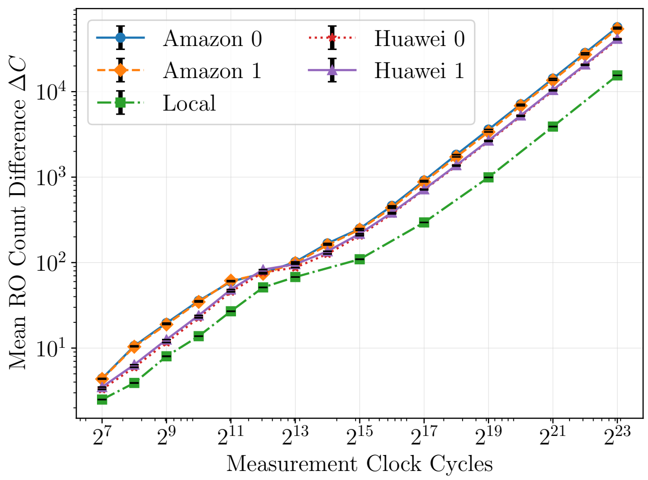


Fig. 14: Average count differences ΔC for different measurement periods 2^t, with 99% confidence intervals.

To account for temperature and voltage fluctuations as well as manufacturing variations, we use a Manchester encoding scheme. With Manchester encoding, a 0-bit is encoded as the pair $(0, 1)$, i.e., the transmitters are disabled for one measurement period, and then enabled for the next one, with a 1-bit reversing this order. Although this effectively halves the bandwidth compared to a simple threshold, it allows for on-chip classification of data, by comparing two successive measurements c_0 and c_1. If $c_0 > c_1$, the bit is classified as a 0, while it is classified as a 1 if $c_0 < c_1$ (we always report equality as an error). The bandwidth b_t of this encoding scheme can be calculated as follows:

$$b_t = \frac{f_c}{2^{t+1}} \qquad (2)$$

where f_c is the clock frequency, and 2^t the measurement period. In the default setup, $t = 7$, so b_t is over 1.17 Mbps for the local VCU118 board, 781 kbps for the Huawei boards, and 488 kbps for the AWS boards, with over 99.9% accuracy.

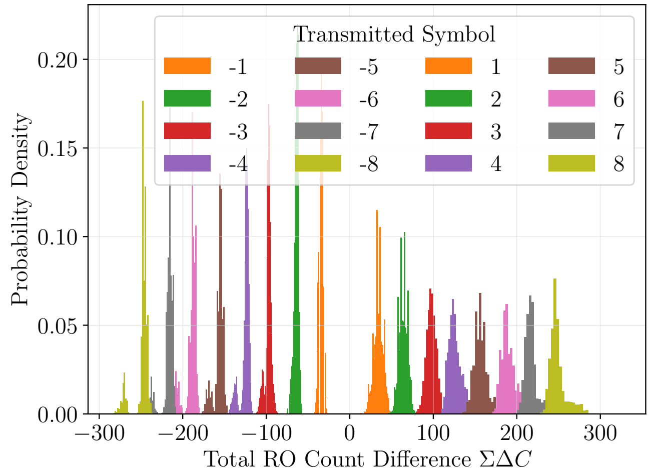


Fig. 15: Histogram of the RO count differences sum $\Sigma\Delta C$ for different symbols. Symbols $\pm i$ correspond to i transmitters, enabled during the first or second measurement periods.

Increasing the measurement time of 2^t clock cycles reduces bandwidth, but increases ΔC linearly, as shown in Figure 14. However, larger count differences can reduce accuracy: errors increase to about 1.1% for $t \geq 15$ on the VCU118 board, as more prolonged environmental fluctuations result in bit flips.

B. *Multi-Bit Transmissions*

Although in the previous sections all of the T transmitters were enabled or disabled simultaneously, in this section we show that we can further increase the bandwidth by selectively enabling only some of the transmitters. Specifically, with T transmitters, we can increase the bandwidth by a factor of $1 + \log_2 T$ compared to the simple Manchester encoding scheme. We encode the $2T$ symbols $\pm 1, \ldots, \pm T$ in two measurement periods as follows: to transmit symbol $1 \leq i \leq T$, we first disable all transmitters during the first measurement period, and then enable i transmitters during the second measurement period. To transmit symbol $-T \leq i \leq -1$, we reverse the process, first enabling i transmitters, and then disabling them.

However, as the count difference ΔC for an individual RO is small, we consider the sum of all such count differences $\Sigma\Delta C$. We plot this sum in Figure 15 for the VCU118 board, where we have increased the number of ROs per transmitter to $N_T = 2{,}000$. Denote the $\Sigma\Delta C$ measurements for i enabled transmitters by the set S_i, and let p_i^α be the α percentile of S_i. We use $t_i^\alpha = (p_{i-1}^{100-\alpha} + p_i^\alpha)/2$ as the lower threshold for symbol i (with $1 \leq i \leq T$), classifying a measurement $s > 0$ as symbol i if $t_i^\alpha \leq s < t_{i+1}^\alpha$ (the $s < 0$ case is analogous for $-T \leq i \leq -1$). Using this classification scheme with $\alpha \in \{1, \ldots, 20\}$, we can recover all $2T$ symbols over 96% of the time, reaching a bandwidth of $b_t^T = \log_2{(2T)} \cdot f_c / 2^{t+1} = 4.6\,\text{Mbps}$. The maximum accuracy of 97.6% occurs for $\alpha = 8$, while $\alpha = 0$, corresponding to minima and maxima, can correctly recover about 80% of transmissions.

VII. Discussion

In this section, we discuss existing defense mechanisms (Section VII-A), limitations of the encoding scheme (Section VII-B), and alternative applications that can make use of cross-SLR information leakage (Section VII-C).

A. *Countermeasures*

Prior work has shown that physical isolation is a necessary prerequisite for secure multi-tenant FPGAs [11]–[13], despite “large costs in terms of frequency and routing congestion” [46]. However, isolation within a die is not a sufficient protection mechanism on its own [24], [25], [30], [47]. This paper has shown for the first time that isolation across SLRs is also not enough to protect multi-tenant FPGA designs. As a result, defense mechanisms must prevent the information leakage either from occurring, or from being detectable.

Prior work has proposed placement and routing restrictions to make some types of information leakage undetectable [11]–[13]. However, no routing constraints are used by our setup, while placement directives are only used to measure the effect of distance on the ensuing channel. As a result, such restrictions are incapable of preventing cross-SLR communication.

Another option to mitigate the channels is to prevent receiver circuits from being instantiated in cloud FPGAs. Simply banning combinatorial loops [14], [15], [24], [36], [47] has proven to be insufficient, since both Time-to-Digital Converters (TDCs) [30], [31], [47] and alternative ring oscillator (RO) designs [13], [33] can bypass design checks implemented by some cloud FPGA providers, with others not deploying any such checks at all. These alternative designs could be detected in some cases, by banning latches [13], [15], [33] and only allowing global clocks to drive flip-flops [13], [33]. However, alternative TDC designs without latches, and gated global clocks through clock enable pins may still be able to circumvent these proposed checks. The design of receivers and defense checks is a cat-and-mouse game, with no clear indication that all receivers can be caught by cloud providers.

Another defense mechanism is to prevent transmitter circuits from being created in cloud FPGAs. Although ROs were used in this paper, other designs with large dynamic power consumption (e.g., switching many Programmable Interconnect Points (PIPs) [48]) can also be used. Consequently, malicious senders can likely still find circuit designs that modulate dynamic power draw despite cloud FPGA design rule checks.

Monitoring unusual power draw activities can also help, but fluctuations can come from legitimate circuits, which the adversary may exploit, for instance to recover cryptographic keys [30], [31], [47]. As a result, hardware changes are needed, such as making the power supplies of different tenants (i.e., different SLR dies) independent. Overall, significant electrical improvements are needed in future FPGA architectures, and will likely come with heavy performance and energy penalties.

B. *Synchronization and Encoding*

Manchester Encoding is often used for its self-clocking properties in covert channels [40] and protocols like

10BASE-T Ethernet [19]. However, in this paper, rising and falling edges are detected through differences in RO frequencies. Absent an external synchronization method, the receiver must sweep the possible clock phases (linearly or with binary search): the largest (average) RO count difference ΔC corresponds to a synchronized receiver and transmitter. Future work could thus examine communication in practice with an unsynchronized channel, and measure bandwidth and accuracy in the presence of third-party activity on the device. How to improve bandwidth and accuracy through error correcting codes, repeated measurements, and alternative aggregation functions (e.g., weighted sums) could also be explored.

C. Alternative Cross-SLR Leakage Applications

Although in this paper we primarily investigated covert communication between users or IP cores of different trust levels, the same mechanism could also be used for side-channel attacks [11], [29], voltage-/fault-attack detection [47], and IP core watermarking [11], [12], [49]. Moreover, although we focused on Virtex UltraScale+ FPGAs, other device generations using SLRs, such as Kintex UltraScale FPGAs on the cloud, are likely also susceptible to the same source of information leakage. Finally, as the root cause of the vulnerability seems to lie in shared power distribution, information leakage between soft and hard cores, for instance on Intel and Xilinx FPGA-CPU hybrids, might also be possible and worth investigating.

VIII. Related Work

This section summarizes prior work on FPGA security, both in the cloud setting (Section VIII-A), and in other remote scenarios without physical access (Section VIII-B).

A. Cloud and Virtualized FPGA Security

Security research on cloud FPGAs has primarily focused on single-tenant applications, e.g., to protect designs from untrusted cloud infrastructures [8]. Other work has conversely investigated how to protect the cloud provider from malicious user logic, suggesting logical and physical isolation, bitstream protection at compilation and deployment, and compile- and run-time checks of user designs [37]. However, with virtualized FPGAs (vFPGAs) and partial reconfiguration gaining traction, many designs have been proposed to accommodate for multi-tenant occupancy of physical FPGA resources [38]. Logical isolation is often a key component of multi-tenant approaches, though physical isolation is not always enforced [6], [7], [39]. Unfortunately, even designs with physical isolation [21]–[23] do not consider or protect against side- or covert-channel attacks, despite their use of "fencing" regions for stricter isolation [37]. As we showed in this paper, even physical isolation along Super Logic Regions on distinct physical dies is not enough to protect against covert-channel communication between vFPGAs.

B. Remote FPGA Attacks

Although traditional covert- and side-channel attacks on FPGAs require physical access to the device [34], [49], temperature- and voltage-based remote FPGA attacks are possible. The works by Iakymchuk et al. [18] and Tian and Szefer [36] lie in the former category, performing temperature-based covert communications within an FPGA and between consecutive users of the same FPGA board respectively. However, these thermal covert channels are slow ($< 1\,\mathrm{bps}$).

By contrast, voltage-based attacks can have much higher bandwidth compared to thermal channels. For example, Giechaskiel et al. identified a crosstalk effect in Xilinx FPGAs due to long-wire capacitive coupling [11], [12], and used it to create a $6\,\mathrm{kbps}$ covert channel. The same phenomenon was then investigated for Intel devices [28], [29], where it was shown that the long-wire leakage can also be used to conduct Differential Power Analysis (DPA) on an AES core, and extract its key. These attacks were performed locally, and with logical, but not physical isolation, unlike our fast ($4.6\,\mathrm{Mbps}$) cloud-based covert channel, which operates under assumptions of physical isolation of logic to separate SLRs.

It should be noted that the switching activity of large logic designs can alternatively be used to cause remote fault attacks on FPGAs [14], [48]. These attacks can crash the FPGA [14], cause timing violations to recover cryptographic keys [24], or bias True Random Number Generators (TRNGs) [25]. Ring Oscillators (ROs) and Time-to-Digital Converters (TDCs) have also been used to conduct intra-chip [25], [30], [47] and inter-chip [31] side-channel attacks. Although some cloud providers, such as Amazon AWS, detect ROs and prohibit their use [2], recent work [13], [33] has demonstrated that alternative RO designs can bypass cloud restrictions. Neither of the two works conducted a practical attack: Sugawara et al. merely proved that instantiation of alternative ROs was possible on AWS [33], while Giechaskiel et al. characterized long-wire leakage on local and cloud FPGAs leveraging ROs that are likewise not caught by the design rule checks of the cloud providers [13]. Long-wire leakage requires transmitter and receiver long wires to be adjacent, whereas we have constructed the first cloud-based covert channel, which remains fast and accurate despite physical isolation to separate SLRs.

IX. Conclusion

In this paper, we showed that cross-SLR covert-channel communication is possible without physical access to or modification of the FPGA boards. We demonstrated a $4.6\,\mathrm{Mbps}$ cross-SLR covert channel with over 97.6% accuracy on five Xilinx Virtex UltraScale+ boards, both locally and on the Amazon and Huawei FPGA clouds. We also characterized the accuracy and bandwidth of the covert communication channel both analytically and experimentally across multiple parameters, such as the locations, types, and sizes of the source transmitters and sink receivers. We finally highlighted the need for hardware-level architectural changes in order to support secure multi-tenant FPGAs, which are currently not possible due to the threats exposed in this work. Python software and pre-compiled Amazon FPGA Images (AFIs) will be open-sourced at https://caslab.csl.yale.edu/code/slr-covert-channel/.

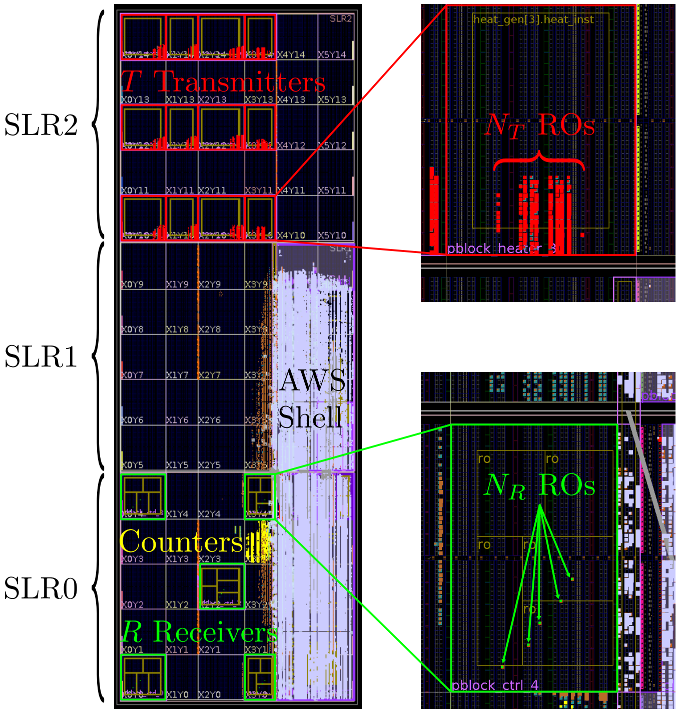


Fig. 16: Vivado screenshot of the experimental setup on AWS. The receivers (green) and counters (yellow) are on SLR 0, while the transmitters (red) on SLR 2. Shell logic (grey) spans SLRs 0 and 1, and interfaces (brown) with the control logic.

Appendix

Figure 16 contains a Vivado screenshot of the measurement architecture on Amazon Web Services (AWS) for one of the experiments of Section V-C. Cross-SLR communication is possible even between SLRs 0 and 2, with minimal loss in accuracy and bandwidth, despite the activity in the cloud shell.

Acknowledgment

This work was supported in part by NSF grants 1716541 and 1901901. We also thank Amazon for donating AWS EC2 research credits.

References

[1] Alibaba Cloud, "Elastic compute service: Instance type families," https://www.alibabacloud.com/help/doc-detail/25378.htm#f1, Accessed: 2019-09-13.

[2] AWS GitHub, "AWS EC2 FPGA HDK+SDK errata," https://github.com/aws/aws-fpga/blob/master/ERRATA.md, Accessed: 2019-09-13.

[3] AWS News Blog, "Developer preview – EC2 instances (F1) with programmable hardware," https://aws.amazon.com/blogs/aws/developer-preview-ec2-instances-f1-with-programmable-hardware/, Accessed: 2019-09-13.

[4] Baidu Cloud, "FPGA cloud compute," https://cloud.baidu.com/product/fpga.html, Accessed: 2019-09-13.

[5] C. Bobda, J. Mead, T. J. Whitaker, C. Kamhoua, and K. Kwiat, "Hardware sandboxing: A novel defense paradigm against hardware trojans in systems on chip," in *International Symposium on Applied Reconfigurable Computing (ARC)*, 2017.

[6] S. Byma, J. G. Steffan, H. Bannazadeh, A. L. Garcia, and P. Chow, "FPGAs in the cloud: Booting virtualized hardware accelerators with OpenStack," in *Field-Programmable Custom Computing Machines (FCCM)*, 2014.

[7] F. Chen, Y. Shan, Y. Zhang, Y. Wang, H. Franke, X. Chang, and K. Wang, "Enabling FPGAs in the cloud," in *Conference on Computing Frontiers (CF)*, 2014.

[8] K. Eguro and R. Venkatesan, "FPGAs for trusted cloud computing," in *Field Programmable Logic and Applications (FPL)*, 2012.

[9] E. El-Araby, I. Gonzalez, and T. El-Ghazawi, "Virtualizing and sharing reconfigurable resources in high-performance reconfigurable computing systems," in *High-Performance Reconfigurable Computing Technology and Applications (HPRCTA)*, 2008.

[10] R. Elnaggar, R. Karri, and K. Chakrabarty, "Multi-tenant FPGA-based reconfigurable systems: Attacks and defenses," in *Design, Automation & Test in Europe (DATE)*, 2019.

[11] I. Giechaskiel, K. Eguro, and K. B. Rasmussen, "Leakier wires: Exploiting FPGA long wires for covert- and side-channel attacks," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 12, no. 3, pp. 11:1–11:29, Sep 2019.

[12] I. Giechaskiel, K. B. Rasmussen, and K. Eguro, "Leaky wires: Information leakage and covert communication between FPGA long wires," in *Asia Conference on Computer and Communications Security (ASIACCS)*, 2018.

[13] I. Giechaskiel, K. B. Rasmussen, and J. Szefer, "Measuring long wire leakage with ring oscillators in cloud FPGAs," in *Field Programmable Logic and Applications (FPL)*, 2019.

[14] D. R. E. Gnad, F. Oboril, and M. B. Tahoori, "Voltage drop-based fault attacks on FPGAs using valid bitstreams," in *Field Programmable Logic and Applications (FPL)*, 2017.

[15] D. R. E. Gnad, S. Rapp, J. Krautter, and M. B. Tahoori, "Checking for electrical level security threats in bitstreams for multi-tenant FPGAs," in *Field-Programmable Technology (FPT)*, 2018.

[16] A. Hajimiri, S. Limotyrakis, and T. H. Lee, "Jitter and phase noise in ring oscillators," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 34, no. 6, pp. 790–804, Jun 1999.

[17] T. Huffmire, B. Brotherton, T. Sherwood, R. Kastner, T. Levin, T. D. Nguyen, and C. Irvine, "Managing security in FPGA-based embedded systems," *IEEE Design & Test of Computers (D&T)*, vol. 25, no. 6, pp. 590–598, Nov-Dec 2008.

[18] T. Iakymchuk, M. Nikodem, and K. Kępa, "Temperature-based covert channel in FPGA systems," in *Reconfigurable Communication-Centric Systems-on-Chip (ReCoSoC)*, 2011.

[19] IEEE Standard for Information Technology, "Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," *IEEE Standards*, vol. IEEE 802.3–2002, Mar 2002.

[20] Intel IT Peer Network, "Intel processors and FPGAs—better together," https://itpeernetwork.intel.com/intel-processors-fpga-better-together/#gs.eeqr8x, Accessed: 2019-09-13.

[21] A. Khawaja, J. Landgraf, R. Prakash, M. Wei, E. Schkufza, and C. J. Rossbach, "Sharing, protection, and compatibility for reconfigurable fabric with AMORPHOS," in *Operating Systems Design and Implementation (OSDI)*, 2018.

[22] O. Knodel, P. R. Genssler, F. Erxleben, and R. G. Spallek, "FPGAs and the cloud – An endless tale of virtualization, elasticity and efficiency," *International Journal on Advances in Systems and Measurements*, vol. 11, no. 3-4, pp. 230–249, 2018.

[23] O. Knodel, P. R. Genssler, and R. G. Spallek, "Virtualizing reconfigurable hardware to provide scalability in cloud architectures," in *International Conference on Advances in Circuits, Electronics and Micro-electronics (CENICS)*, 2017.

[24] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "FPGAhammer: Remote voltage fault attacks on shared FPGAs, suitable for DFA on AES," *Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2018, no. 3, pp. 44–68, Sep 2018.

[25] D. Mahmoud and M. Stojilović, "Timing violation induced faults in multi-tenant FPGAs," in *Design, Automation & Test in Europe (DATE)*, 2019.

[26] Microsemi Corporation, "RISC-V CPUs," https://www.microsemi.com/product-directory/mi-v-embedded-ecosystem/4406-risc-v-cpus, Accessed: 2019-09-13.

[27] G. Provelengios, D. Holcomb, and R. Tessier, "Characterizing power distribution attacks in multi-user FPGA environments," in *Field-Programmable Logic and Applications (FPL)*, 2019.

[28] G. Provelengios, C. Ramesh, S. B. Patil, K. Eguro, R. Tessier, and D. Holcomb, "Characterization of long wire data leakage in deep submicron FPGAs," in *Field-Programmable Gate Arrays (FPGA)*, 2019.

[29] 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 *Field-Programmable Custom Computing Machines (FCCM)*, 2018.

[30] F. Schellenberg, D. R. E. Gnad, A. Moradi, and M. B. Tahoori, "An inside job: Remote power analysis attacks on FPGAs," in *Design, Automation & Test in Europe (DATE)*, 2018.

[31] ——, "Remote inter-chip power analysis side-channel attacks at board-level," in *International Conference on Computer-Aided Design (ICCAD)*, 2018.

[32] D. M. Shila, V. Venugopalan, and C. D. Patterson, "Unraveling the security puzzle: A distributed framework to build trust in FPGAs," in *International Conference on Network and System Security (NSS)*, 2015.

[33] T. Sugawara, K. Sakiyama, S. Nashimoto, D. Suzuki, and T. Nagatsuka, "Oscillator without a combinatorial loop and its threat to FPGA in data centre," *Electronics Letters*, vol. 15, no. 11, pp. 640–642, May 2019.

[34] J. Sun, R. Bittner, and K. Eguro, "FPGA side-channel receivers," in *Field-Programmable Gate Arrays (FPGA)*, 2011.

[35] Tencent Cloud, "Cloud virtual machine instance types," https://intl.cloud.tencent.com/document/product/213/11518#FX2, Accessed: 2019-09-13.

[36] S. Tian and J. Szefer, "Temporal thermal covert channels in cloud FPGAs," in *Field-Programmable Gate Arrays (FPGA)*, 2019.

[37] S. Trimberger and S. McNeil, "Security of FPGAs in data centers," in *International Verification and Security Workshop (IVSW)*, 2017.

[38] A. Vaishnav, K. D. Pham, and D. Koch, "A survey on FPGA virtualization," in *Field Programmable Logic and Applications (FPL)*, 2018.

[39] J. Weerasinghe, F. Abel, C. Hagleitner, and A. Herkersdorf, "Enabling FPGAs in hyperscale data centers," in *International Conference on Ubiquitous Intelligence and Computing, Autonomic and Trusted Computing, Scalable Computing and Communications (UIC-ATC-ScalCom)*, 2015.

[40] Z. Wu, Z. Xu, and H. Wang, "Whispers in the hyper-space: High-bandwidth and reliable covert channel attacks inside the cloud," *IEEE/ACM Transactions on Networking (TNET)*, vol. 23, no. 2, pp. 603–615, Apr 2015.

[41] Xilinx, Inc., "Large FPGA methodology guide, including Stacked Silicon Interconnect (SSI) technology (UG872)," https://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/ug872_largefpga.pdf, Accessed: 2019-09-13.

[42] ——, "UltraScale architecture and product data sheet: Overview (DS890)," https://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf, Accessed: 2019-09-13.

[43] ——, "UltraScale architecture configuration: User guide (UG570)," https://www.xilinx.com/support/documentation/user_guides/ug570-ultrascale-configuration.pdf, Accessed: 2019-09-13.

[44] ——, "Virtex-6 family overview (DS150)," https://www.xilinx.com/support/documentation/data_sheets/ds150.pdf, Accessed: 2019-09-13.

[45] ——, "Xilinx powers Huawei FPGA accelerated cloud server," https://www.xilinx.com/news/press/2017/xilinx-powers-huawei-fpga-accelerated-cloud-server.html, Accessed: 2019-09-13.

[46] S. Yazdanshenas and V. Betz, "Interconnect solutions for virtualized field-programmable gate arrays," *IEEE Access*, vol. 6, pp. 10 497–10 507, Feb 2018.

[47] M. Zhao and G. E. Suh, "FPGA-based remote power side-channel attacks," in *IEEE Symposium on Security and Privacy (S&P)*, 2018.

[48] K. M. Zick, M. Srivastav, W. Zhang, and M. French, "Sensing nanosecond-scale voltage attacks and natural transients in FPGAs," in *Field-Programmable Gate Arrays (FPGA)*, 2013.

[49] D. Ziener, F. Baueregger, and J. Teich, "Using the power side channel of FPGAs for communication," in *Field-Programmable Custom Computing Machines (FCCM)*, 2010.