DoS-FPGA: Denial of Service on Cloud FPGAs via Coordinated Power Hammering

Hassan Nassar*, Philipp Machauer, Lars Bauer, Dennis Gnad*, Mehdi Tahoori* and Jörg Henkel*
*Karlsruhe Institute of Technology (KIT), Institute for Computer Engineering (ITEC), Germany
*Email: {hassan.nassar, dennis.gnad, mehdi.tahoori, henkel}@kit.edu

ABSTRACT

The adoption of FPGA instances by major cloud service providers (CSPs) reflects the growing demand for accelerated and heterogeneous computing across various applications, e.g., AI. To improve the efficiency, utilization and virtualization, multi-tenant FPGAs allow multiple users to utilize FPGA resources concurrently, with each FPGA partition assigned to a separate tenant. However, this introduces significant security vulnerabilities, such as the potential for attacks by configuring a malicious circuit in one tenant's FPGA partition. One notable vulnerability is disrupting the FPGA's power distribution network, leading to faults or even crashing the entire FPGA, affecting other tenants. Usually, such an attack requires a considerable amount of resources. A naive solution would be splitting an FPGA into smaller fractions to reduce the potential for successful Power-Hammering by individual tenants and enhance the security. However, our paper demonstrates that even with smaller fractions per tenant, attacks can still occur. We propose the threat of coordinated attacks, where malicious tenants use an unintended covert channel between them. We practically validate this threat in a real cloud computing environment by introducing a synchronized and coordinated power-hammering attack from multiple malicious tenants. These tenants synchronize their actions using a voltage-based covert channel. Our results reveal the success of the attack, surpassing state-of-the-art countermeasures and detection mechanisms with a success rate exceeding 90%, compared to 30% for uncoordinated attacks.

CCS CONCEPTS

• **Security and privacy → Hardware attacks and countermeasures**.

KEYWORDS

FPGA, Hardware Security, Multi-tenant FPGAs

This work was partially funded by the "Helmholtz Pilot Program for Core Informatics (kikit)" at Karlsruhe Institute of Technology and by the German Federal Ministry of Education and Research (BMBF) through grant 01IS23066 as part of the Software Campus Project "HE-Trust". This work was supported in part by AMD under the Heterogeneous Accelerated Compute Clusters (HACC) program.

ICCAD '24, October 27–31, 2024, New York, NY, USA

ACM ISBN 979-8-4007-1077-3/24/10.
https://doi.org/10.1145/3676536.3676843

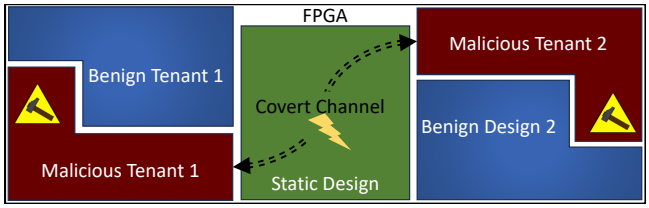

Figure 1: System overview. Multiple attacking tenants, at least two, coordinate their power-hammering attack via covert channel intra-chip communication. This leads to successful power-hammering and causes faults to the benign tenants.

1 INTRODUCTION

Modern computing systems are often shared between different users and applications. By sharing resources in cloud environments, their utilization can be increased, optimizing availability, energy efficiency, and scalability [1]. To securely provide the necessary multi-user capabilities, various isolation mechanisms are introduced, for which various security assumptions are considered. Covert channels are one of the methods that can be used to exfiltrate information from one isolated security context to another, and can quickly become a threat to overall systems security [2–7]. One of the most prominent examples of massive security breaches in mainstream hardware is cache covert channels that are used in speculative execution attacks [8].

FPGAs are gaining more importance in modern computing systems. Their versatility and adaptability allow the optimization of a wide array of applications. FPGAs can be employed for accelerating complex algorithms, offering significant speedup over traditional CPUs [9], and are among the most efficient accelerators for machine learning inference [10]. Cloud Service Providers (CSPs) allow users to accelerate arbitrary applications on their FPGA cloud infrastructures [1, 11–13], while on the other side, even edge and IoT applications can benefit from FPGA acceleration. Specifically, multi-tenant FPGAs are a concept that can allow CSPs to provide their services more efficiently [14]. In such a case, the FPGA is partitioned into multiple regions, each of them assigned to a different user. In this manner, the higher utilization can increase throughput at the same fixed energy budget, increasing efficiency. From an economic point of view, virtualization and multi-tenancy in FPGAs help in making more profit using idle compute resources and follow the general trend of cloud setups.

However, in contrast to multi-tenancy that is already deployed for classical computing devices or GPUs, multi-tenancy for FPGAs faces several challenges [15, 16]. One of the main challenges is the security concerns related to multi-tenancy in FPGAs. Among others, these security concerns include the threat of the so-called power-hammering attacks [15, 17]. These attacks induce disturbances in the power distribution network by creating momentous high peaks

of power usage. By controlling the frequency of producing these peaks, the attacker can inject faults or crash the FPGA.

Since the emergence of these attacks, several countermeasure and detection tools have been proposed [18–21]. However, stealthier attacks are continuously being proposed to escape such countermeasures [21–24]. These stealthier attacks are often weaker than traditional attacks and need more FPGA resources to be successful [19]. This would require having the malicious tenants occupy a large portion of the FPGA, which can reveal their maliciousness.

In this work, We take a different yet novel approach. We launch the attack from multiple small tenants implementing stealthy attacks. The stealthiness means they are hard to detect by existing tools [25] but also means that the resource utilization of each malicious tenant alone is not sufficient for a power-hammering attack. Therefore, the goal is that they together create enough disturbance to the power distribution network and lead to a successful attack. As Fig. 1 shows, to coordinate the attack from the multiple tenants we use a covert channel based on modulating the communication to the on-chip power distribution network. Within the several tenants on the FPGA, it is required that at least two of them to be malicious. In summary, our novel contributions are:

- We are the first to address coordinating power-hammering attacks among multiple weak and small malicious tenants in multi-tenant FPGAs, resulting in a powerful yet stealthy attack.
- We are the first to show that covert channels can be used to coordinate power-hammering attacks in multi-tenant FPGAs.
- We show how our coordinated power-hammering attack bypasses state-of-the-art countermeasures.
- We show how our attack can be deployed on a real cloud infrastructure.

The rest of the paper is organized as follows: Section 2 gives the necessary background on our work. Section 3 explains our novel attack and how it functions. We implement our attack in Section 4 and evaluate it in Section 5. We discuss the attack, its limitations, and possible countermeasures in Section 6. Finally, we draw conclusions in Section 7.

2 BACKGROUND

As the computing landscape evolves towards more heterogeneity, FPGAs are assuming an important role. They have transitioned from being standalone devices to being seamlessly integrated by major processor manufacturers like AMD and Intel into MPSoCs. These advanced chips combine processor cores with the flexibility and reconfigurability of FPGAs.

FPGAs are being more integrated into cloud infrastructures, and most of major cloud service providers have acceleration solutions using FPGAs. Multi-tenant FPGAs is a topic of high interest for both academia [1] and industry, e.g., IBM [11]. It accommodates different users, those with high computing needs and those with low computing needs, and amortizes the costs. The idea is to have powerful FPGAs on the cloud and provide multi-tenancy to keep the FPGA area fully utilized to amortize the costs. This is relevant due to the increasingly large FPGAs available, where typical users, i.e., tenants, often do not require the full capacity. To boost efficiency, FPGAs are shared among tenants, divided into static sections for managing communication and dynamic sections with multiple accelerator slots. Tenants can then load custom accelerators into these slots. However, multi-tenant FPGAs remain largely unimplemented in commercial platforms, and among the reasons are the security concerns [16, 26].

2.1 Voltage-based Attacks on Multi-tenant FPGAs

Multi-tenant FPGAs raise significant security concerns [16, 26]. Of particular concern are physical attacks that do not require direct physical access to the FPGA chip [17, 27–30]. Power-hammering attacks involve the deliberate use of high-power-consuming logic, so-called "power wasters" that are implemented using legitimate FPGA logic, to disrupt the power distribution network within the FPGA, which can cause faults or lead to the complete crash of the FPGA or its power supply, necessitating manual power cycling for system recovery [17, 30, 31].

The attack runs by continuously enabling and disabling the power wasters, where the outcome of the attack is based on the frequency with which they get enabled and disabled. Depending on the specific FPGA, usually very low frequencies (less than few 10 kHz) have no effect. When increasing the frequency (some 10 −100 kHz ranges), first crashes become possible, where an increase can prevent crashing but instead cause computational faults. At very high frequencies, usually no disturbances occur [19, 32].

Such attacks result in substantial availability loss, coming with significant financial losses for Cloud Service Providers (CSPs), as demonstrated in [15]. Table 1 summarizes the attacks and their types. These fault attacks have evolved from using obvious malicious circuits like ring oscillators to more stealthy variants, such as obfuscated ROs [21, 31], short-circuits through specific Block RAM access patterns [22], and by now even benign synchronous IP modules can be used to launch attacks [23, 33].

Table 1: Different types of power-hammering attacks. Several FPGA primitives and circuit designs are used to induce faults and crash the FPGA through power-hammering. The stealthy attacks like Latch-based ones require at least 30% of FPGA resources for a successful attack [19].

Ring Oscillators	Glitch amplification	Sequential Logic
Latch-based [15, 21]	BRAM-based [22]	ISCAS-based [20]
Mux-based [15, 21]	Flip-Flop-based [15]	Register-based [23]
LUT-based [17, 21]	LUT-based [15]	AES-based [23]

For example, ref. [23] shows that AES can be used for power-hammering. Using slight modifications as shown in Fig. 2 will create high power disturbance which leads to successful attacks when several of these units run in parallel.

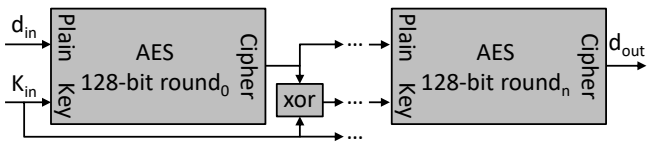

Figure 2: AES-based Power-Hammering attacker from [23]. By xoring the key and the cipher and using a special input pattern and key pattern a power-hammering is successful.

As power-hammering attacks are a threat to cloud-based FPGAs [30, 34] they can also enable side-channel attacks where one tenant attacks the other, by employing various ways of measurement. In FPGAs, it is feasible to subvert the regular FPGA fabric into being used as voltage sensors [27], which can also be done stealthily with regular circuits [24]. That, it is possible, for instance, to extract secret keys from a running cryptographic accelerator in a neighboring victim tenant [35].

2.2 Covert channel Communications on FPGAs

Next to fault and side-channel attacks, several works reported successful covert-channel communications in systems with FPGAs. In [36], it was shown that voltage fluctuations caused by a CPU or GPU in a workstation system can generally be observed in an FPGA PCIe accelerator card when sufficient activity on the CPU or GPU is happening. In [37, 38] it was shown that proper data transmission in the range of 4-5 MBit/s is feasible inside a multi-tenant FPGA. The work in [39] shows a successful power-based covert channel for an FPGA-MPSoC, establishing a communication channel between CPU, FPGA and other components.

However, covert channels on FPGAs are not exclusively power-based. In a similar MPSoC, it was also shown that a shared clock tree can lead to covert channels in which the sender modulates a message on the PLL clock frequency under its control, which can then be observed by the respective FPGA fabric [7]. Furthermore, thermal covert channels were shown in time-multiplexed Cloud FPGAs, which can leak information in longer timescales of usage [40].

These covert channel implementations are however not optimal for our attack. The covert channels implemented in [7, 36, 39] do not have both the sender and receiver both on the FPGA logic. The covert channel from [40] targets time-multiplexed FPGAs which is not our system. Both covert channels from [17, 37] are the most similar to us. However, they use LUT-based ROs for the sender. This can be easily detected and stopped by current solutions implemented by Vivado and service providers [15, 21]. Moreover, they focus on high throughput which is not needed for our synchronization and the sender is active for a longer period which makes it easier for online countermeasures to detect it [18].

2.3 Countermeasures against Power-Hammering Attacks

Several countermeasures against power-hammering attacks in FPGAs exist; they can be offline or online. Offline countermeasures try to detect malicious designs before loading the tenant-specific accelerators [20, 21]. Online countermeasures on the other hand try to detect and/or stop malicious tenants once they start their malicious activity [18, 19].

Both types of countermeasures have their pitfalls. First, offline countermeasures are never 100% accurate in detection [41], therefore, they cannot reliably stop all attacks. Moreover, new attacks are continuously evolving and they are built to escape detection from the current offline countermeasures. For example, the RAM-based attack [22] would not be caught by the detection tool from [21] as it is a different type of attack than what the tool is built for. This is particularly harder with new attacks like the AES-based ones. Furthermore, offline countermeasures scan the bitstream or the netlist to find the malicious circuitry [20], which raises concerns about IP theft of proprietary tenant designs.

Second, for online countermeasures, two solutions exist. LoopBreaker [19] requires the identification of possible malicious tenants using an offline tool, which comes with all the pitfalls of the offline countermeasures. The solution in [18] can successfully locate a malicious tenant. It has two limitations: first, it needs 20 µs to detect an attack. Second, it can only detect attacks running at frequencies of up to 200 kHz. However, successful attacks have been demonstrated at higher frequencies and require significantly less time than 20 µs to be successful.

3 PROPOSED COORDINATED ATTACK

All known attacks thus far are limited to using the resources of a single tenant which as we show in Section 5.3 can be stopped by state-of-the-art countermeasures. For our proposed attack, we use a covert channel to synchronize multiple tenants to go beyond the resource limits of a single tenant. Each tenant is small and uses at maximum 15% of the resources of the FPGA and therefore it is harder to detect it as usually the attacks require around 30% of the FPGA resources as shown for latch-based attacks in [19]. Proper synchronization mechanism is crucial for the success of our attack, without this synchronization, the attack would fail as we show in Section 5.3. Using this strategy, the attack has access to more resources simultaneously. This makes it harder for online countermeasures that are designed to mainly deal with a single malicious tenant active at a time [19].

3.1 System Overview, Assumptions and Goals

The system targeted by our attack is illustrated in Fig. 3. It is a multi-tenant setup where at least two tenants are malicious and all other tenants are victims. The communication works in a bidirectional way. The setup also includes a static region that is designed by the cloud service provider. Among others, the static region serves as a clock source for the different tenants and their communication interface to the outside world. This communication is essential for the tenants so they can get input data or communicate intermediate/final results for the customers renting the area on the FPGA. The static design also ensures no intra-FPGA communication between the tenants.

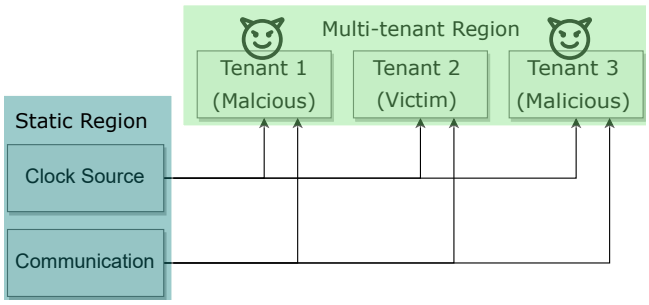

Figure 3: Expanded structural overview for the system targeted by our attack. At least two malicious tenants coordinate the power-hammering via a covert channel. The static region provides the needed infrastructure to all tenants.

We assume that each malicious tenant knows how many other malicious tenants are needed for a successful attack. However, other than having the same clock, the malicious tenants do not have any other means of synchronization nor can they know whether or not they reside on the same FPGA with other malicious tenants or the number of the malicious tenants on the FPGA. Therefore, we aim to establish a covert channel to coordinate the attack reliably. As we are performing a power-based attack, it is sensible to use the same infrastructure to establish a power-based covert channel. We aim for the covert channel to be able to bypass state-of-the-art countermeasures [18, 19].

By observing the characteristics of these countermeasures, we realized that they react well (i) if the disturbance runs continuously for a period in the range of microseconds and (ii) if the disturbance is strong enough in order not to have false positives and mistakenly stop non-malicious tenants. Therefore, we aim to have the attack (i) use short cycles when sending data and (ii) create a low yet distinguishable disturbance to the power of the FPGA. Moreover, we aim to have a simple and robust communication protocol. We do not aim to use any sophisticated error correction or modulation schemes to ensure that our receiver is as lightweight as possible.

3.2 Covert Communication Protocol

The communication protocol is simple. It has n communication slots for an attack by n tenants with one slot per tenant. Each slot consists of 150 cycles at 100 MHz, the sending tenant generates a short voltage peak of 10 cycles, waits for 40 cycles, and repeats the same pattern. This is followed by staying silent for 50 cycles. Note that the number of cycles for the pulse and the frequency it is generated with are both tailored to escape detection by countermeasures from Section 2.3.

This design is simple yet effective. Generating two voltage peaks (which are sensed by the listening malicious tenants) act as synchronization. If the other malicious tenants do not exactly detect two peaks, they understands that this is not real communication but rather random noise and no synchronization occurs. The silence for 50 cycles serves to increase the resilience to noise. It ensures that the two previous peaks are not just part of some ongoing noisy pattern detected mistakenly. This is important as the existence of several victim tenants may coincidentally produce such a pattern.

Each peak is followed by 40 cycles of silence. This design choice compensates for the fact that the sensors detecting the peaks can have offsets and delays. Therefore, using tight windows would lead to undesirable high error rates. Please note that the communication is still at a much lower activity level and not detectable as a fault.

3.3 Design of the Malicious Tenants

The attacker has three main tasks: (i) send covert messages, (ii) receive covert messages, and (iii) run the attack. To perform all three tasks it uses the design from Fig. 4. As mentioned above, the attacker sends peaks via the covert channel. Therefore, the attacker uses a small subset of the "power wasters" (see Section 2.1) to send the peaks. The rest is used for the attack itself. Unlike previous works, our sender uses seemingly benign "power wasters" and successfully modulates them. By continuously enabling and then disabling the power wasters, the attacker causes sufficient power disturbances for communication.

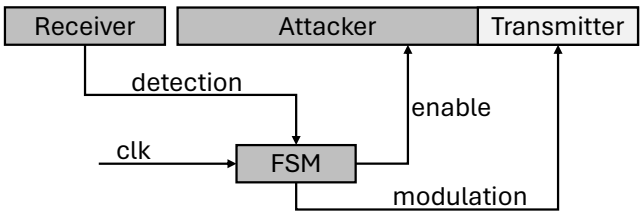

Figure 4: Figures of the design inside the two communication tenants with the connections to the state machine

For the receiving, it is necessary to detect the generated pattern reliably from the other tenants. As we are using a covert channel for communication, there is no direct way to receive this pattern. Therefore, we implement the receiver to detect the pattern generated by the malicious tenants. This pattern causes power disturbance which affects the timing of the circuits implemented on the FPGA. Therefore, the power disturbance can be detected by measuring the speed of a circuit. In order to detect these changes we implement Time to Digital Converters (TDCs) as a cascade of buffers. Each buffer has a delay, and in case power disturbances exist, the delay of each buffer increases. We send a pulse into the TDC and measure how long it needs to go through all buffers. When high power disturbance exists, the signal takes longer than expected to traverse all buffers. Hence, by monitoring the speed of signals traversing the TDC, we can detect the peaks reliably. In total, seven TDCs are used to enable reliable detection via majority voting. The TDCs are designed to be lightweight and cause no significant overhead.

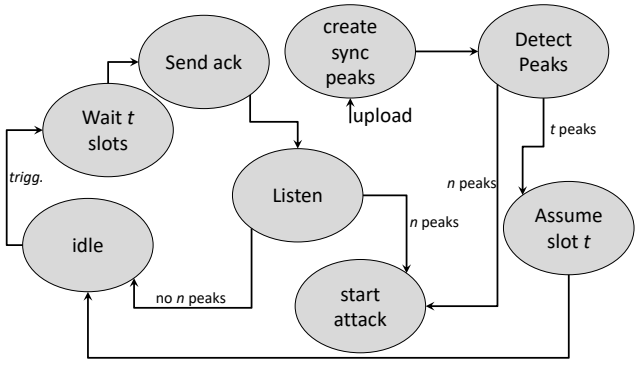

Figure 5: State machine controlling the malicious tenant.

Creating the covert message, figuring out the proper communication slot, and receiving is controlled via the state machine shown in Fig. 5. When a malicious tenant is uploaded, it does not know if other malicious tenants reside on the FPGA or not. Therefore, it directly starts sending the covert message. If it receives the correct number n messages from n tenants, this means that all needed tenants are there and the attack can start right away. However, if it receives less messages of number t it knows that not all the needed tenants are there and it waits for their upload. Based on the number t it assigns itself the appropriate slot in upcoming communications. For example, if it is the first malicious tenant to be uploaded, it will

receive $t = 0$ messages. Therefore, it will self-assign slot 0. This stacks nicely and in a conflictless manner.

If the attack does not start right away the FSM stays in the idle phase and continuously checks for the power disturbance via the TDCs. Once it detects the first peak meaning that a new malicious tenant is uploaded it waits for t slots and then sends its message on its correct slot. If all n messages are received, it starts the attack. Otherwise, it aborts and starts from idle again.

4 EXPERIMENTAL SETUP

4.1 Implementation on the ZCU102 Board

We implement the system from Section 3.1 on a Xilinx ZCU102 FPGA development board containing an UltraScale+ Zynq MP-SoC. The synthesis, placement, and routing are all performed using Vivado 2022.2 running on an Intel i9-12900 system with 64 GiB of memory. A complete run starting from synthesis to bitstream generation takes, depending on the system and the configuration, between 30 and 45 minutes.

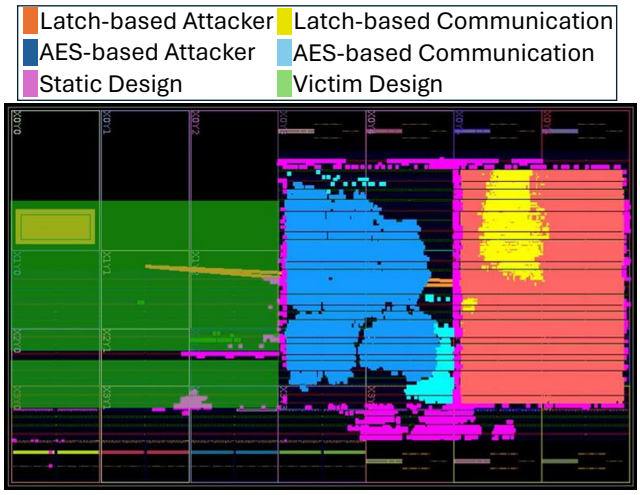

Figure 6: Floorplan of the implemented system with the malicious tenants on the ZCU102 FPGA development board. In this floorplan, a latch-based attack and an AES-based attack are used as the malicious tenants.

The implemented system has three tenant slots alongside static design as Fig. 6 shows. Two slots are for malicious tenants and the final one is for a victim tenant. The sizes for each tenant slot are outlined in Table 3. For the victim tenant, we implement three different designs based on circuits from three benchmarks, ISCAS [42], Groundhog [43], and Berkeley [44]. The circuits used from each benchmark are stated in Table 2

Table 2: Circuits used from each benchmark as background applications to evaluate the robustness of the covert communication.

ISCAS	Groundhog	Berkeley
s208_1	GH09.B.1	ucsb_152_tap_fir–0
s420_1	GH09.B.2	ava–1
s526n	GH09.B.4	top_rs_decode–3
s9234_1	GH09.B.6	uoft_raytracer–3

For malicious tenants, we implement four different designs as the attack part. We use self-oscillating latch-based attacks from [21], AES-based attacks from [23], DES-based, and SHA-based attacks from [25]. All these attacks can bypass any offline checks done by commercial service providers [15]. We also verify that they bypass the offline checks on the AMD Heterogeneous Accelerated Computing Clusters (HACC) [45] as we show below in Section 4.2. Moreover, as three of the attacks are based on benign circuits (AES, DES, and SHA) it will be even harder to detect them by most of the offline detection tools.

Table 3: Size of tenant slots in three tenant setup. The malicious tenant slots are given more resources to be able to fine-tune the attacks. The actual attacks require less than 15% per tenant as we show in Section 5.4

Tenant	malicious 1	malicious 2	Victim
Size (LUTs)	69520	72048	34152
% of LUTs	27.8%	28.8%	13.7%

For the receiver part of the malicious tenants we implement it using TDCs. Figure 7 shows the design of the TDCs. We use the same design from [17] Each TDC is designed as a long delay chain and uses 8 carry chains and 77 registers. The 7 TDCs increase the robustness of the communication and this design allows to have them with a low overhead. The receiver including TDCs uses only 0.6% of the available FPGA resources which is negligible.

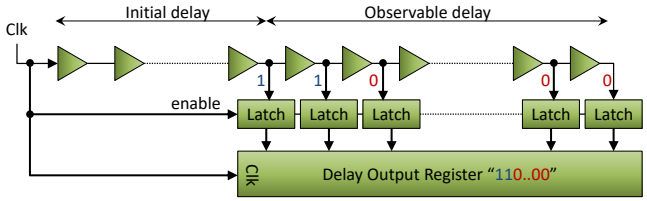

Figure 7: The Time to Digital Converter implemented as a long delay chain based on the implementation from [17].

4.2 Implementation on HACC Cloud Setup

We further extend our evaluation to the Heterogeneous Accelerated Computing Clusters (HACC) [45]. HACC is a cloud setup hosted at several universities and run by AMD. It contains servers accelerated with cloud FPGA boards from Xilinx, e.g., Alveo U200. It natively supports only single-tenant designs. However, it is possible to have the FPGA partitioned into reconfigurable regions to mimic the multi-tenant scenario. We target the Alveo U200 FPGAs in their infrastructure. It is significantly bigger than the ZCU102 FPGA with almost 10× the resources. Therefore, we can divide the FPGA into 15 tenant regions, each with 6% of the FPGA resources (roughly the same size as 60% of the ZCU102) while keeping 10% for the static design.

Figure 8 shows our final design. We assign 10 tenant regions to malicious tenants while keeping 5 for victim tenants. For all four attack designs, we were able to synthesize, generate the bitstream, and upload it to the Alveo U200 within the HACC environment. This proves that the offline checks performed by AMD in the cloud setup does not catch these stealthy attacks. To avoid any financial impact, we did not enable the full attacks on the Alveo cards, but rather only evaluated the covert communication. Moreover, the AMD HACC shows its traffic (how many single-tenants are active

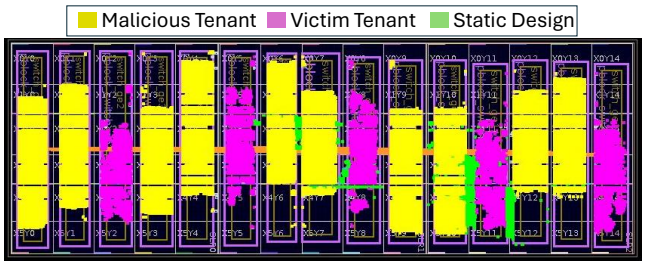

Figure 8: Floorplan of the implemented system on the Alveo U200 board on the HACC cloud. Up to ten malicious tenants residing on the same FPGA. All four attacks from the paper were successfully uploaded to the FPGA.

at any moment) to its users. We use this traffic data, to emulate a multi-tenant scenario where five tenants have to share an FPGA to evaluate whether having an attack from multiple tenants is possible or not in Section 5.5.

5 EVALUATION

5.1 Attack Performance

Based on the system from Section 4.1, we evaluate the success of our attack. We sweep over all the combinations of using each attack together. Moreover, the designs for the victim tenant (the benign designs from the benchmarks) are used to evaluate the effect of having noise from other tenants. We run each unique setup 1000 times to get the results and evaluate how effective our attack is.

Once the two attackers synchronize, they launch the attack. As shown in Section 2, online detection and countermeasures exist. We implement our attack to use a faster attack frequency than [18] can maximally detect (see Sections 2.3 and 3). However, LoopBreaker [19] can be very fast in stopping a malicious tenant if it is targeting the correct tenant before the attack starts.

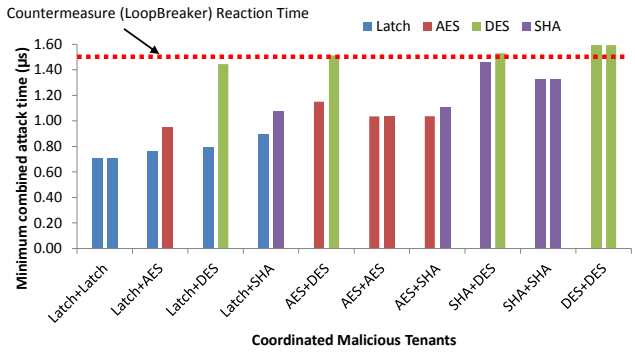

Figure 9: In the presence of a countermeasure that quickly stops one of the two coordinated malicious tenants, this figure shows how long both tenants need to be active together before the not-stopped tenant can successfully finish the attack alone. The bar is plotted for the not-stopped tenant, any bar below the dashed line evades detection.

But even LoopBreaker can only stop one of our two coordinated attacker tenants. Therefore, we need our coordinated attack to create enough disturbance over the PDN before the countermeasure can stop one of the two tenants. In this case, the PDN is weak enough such that the continued disturbance from the remaining tenant alone can crash it. That is, the coordinated attackers need a minimum *combined attack time*, after which one of the tenants can finish the attack alone. For every combination of attackers, Fig. 9 shows the minimal time that both attacker tenants have to run together. The attack will be successful even if the other attacker tenant is stopped immediately after the minimal combined attack time. The results show that Latch-based attacks need the least amount of combined attack time and DES-based attacks need the longest combined attack time. For DES-based attacks, it is often even longer than the reaction time of LoopBreaker, i.e., LoopBreaker could successfully prevent DES-based attacks, if it was targeting the attack partner of DES before the attack started and left DES to continue the attack alone.

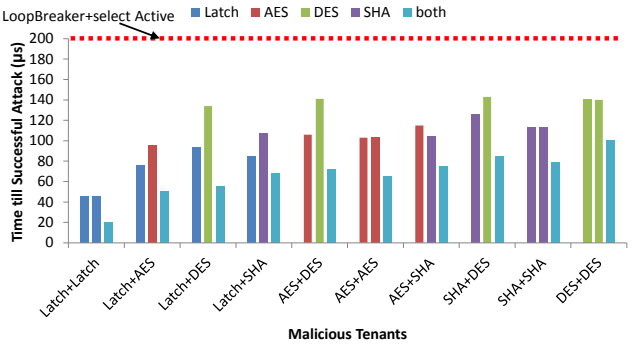

Figure 10: The time needed by attackers after the minimal combined attack time to lead to a successful attack. The red horizontal line shows the time at which the countermeasure could have re-targeted and stopped the second attacker. The results show that all attacks are successful after the minimal combined attack time.

We further extend our evaluation to the case where the countermeasure will stop both tenants sequentially and not only one of them. In such a case, the first attacker-tenant will be stopped very fast and then the other one is stopped slowly (because the countermeasure first needs to re-target the second attacker). The results are shown in Fig. 10. For each combination of attackers, we have three bars, one bar per attacker to see how long it needs to perform a successful attack, and a third bar that shows the result if no countermeasure was used. The results show that for all cases, the attack is faster than the theoretical limit needed by the countermeasure. Therefore, if both attacker tenants can execute for the minimal combined attack time, then the attack will always be successful afterwards.

5.2 Evaluation against Countermeasures

We do not rely only on the theoretical comparison with the speed of the countermeasure. As the LoopBreaker [19] countermeasure is open source, we integrate it into our system and evaluate its effectiveness in stopping our attack. In this experiment, we have LoopBreaker always pointing to one malicious tenant, i.e., we give it a significant advantage (it cannot point at two tenants at the same time). Therefore, if LoopBreaker detects the communication itself, it stops the attack during the communication phase before the real attack even starts. Note that the attack would fail in this experiment for one of three reasons; (i) the communication between

the malicious tenants fails, (ii) the communication was detected by the countermeasure and stopped the malicious tenant it targets, or (iii) the countermeasure stops the targeted malicious tenant before the minimum combined attack time is completed.

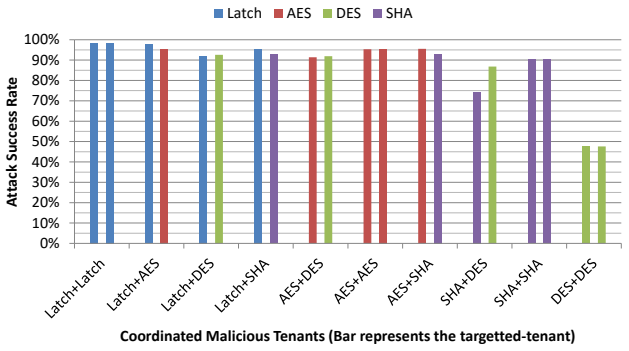

Figure 11: The attack success rate for the different tenant combinations. The countermeasure tries to stop the targeted tenant as fast as possible once it detects malicious activity.

Figure 11 shows the results of the experiment. For most of the cases, the attack is successful with a good percentage (higher than 90%). The main outlier is observed when a DES-based attacker is used. The attack degrades below 90% and even below 80%. It gets significantly worse (below 50%) when a DES-based attacker is used. This is in line with the previous results from Fig. 9 where a DES-based attacker was always performing worse than other attacker circuits.

5.3 Failure of Single-tenant and uncoordinated Attacks

We do not rely only on theoretical reasoning for the need for coordinated attacks. Instead, Fig. 12 shows the attack success rate for (i) single malicious-tenant attacks and (ii) multiple malicious-tenant attacks that are uncoordinated. The setup to generate these results is the same as explained in Sections 4 and 5.1. This experiment shows that for single-tenant attacks, they are not strong enough to cause any successful attack, and countermeasures from State-of-the-art are sufficient against them. For the uncoordinated attacks, the success is a bit higher but does not exceed 30%. Without coordination, the attackers are only active at the same time in a random manner which significantly affects the success rate.

5.4 Resource Utilization

For resource utilization, the upper bound for each tenant is the LUTs available for each tenant slot shown in Table 3. However, we aim to have the utilization as low as possible so it fits in smaller tenants if needed. As long as the communication and the attack are successful we do not increase the resource utilization.

Table 4 shows the resource utilization of our system. The resource utilization for each attacker is between 30.000 LUTs to 37.000 LUTs. We use one sub-part of the attacker for the communication and it requires at most 19.000 LUTs. The benign tenants are in a similar range to the attacker size at 30.000 LUTs. As they are benign, they have no resource utilization for the covert channel. Finally,

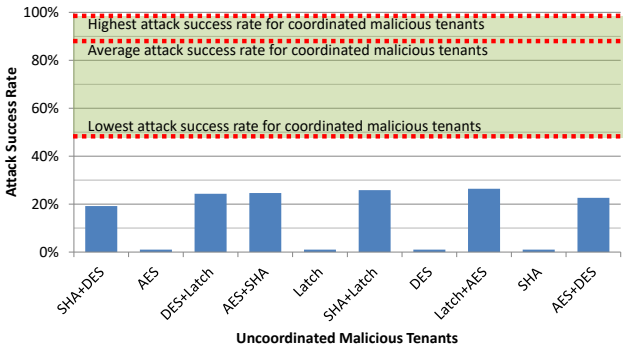

Figure 12: Attack success rate of uncoordinated malicious tenants is less than 30% and for single malicious tenants is 0%. shaded in green is the success range of our coordinated attack for comparison.

Table 4: Resource utilization per tenant in LUTs. 'Communication size' is the number of resources needed to launch the communication. The 'Receiver' (last row) denotes the size of the FSM and TDCs needed in each malicious design.

Tenant	Total size	% of LUTs	Comm. size
Latch-based	37361	14.9%	8349
AES-based	31830	12.7%	12368
DES-based	36128	14.5%	18496
SHA-based	33725	13.5%	14657
Berkeley [44]	28347	11.3%	N.A.
Groundhog [43]	29540	11.8%	N.A.
ISCAS [42]	31825	12.7%	N.A.
Receiver	1449	0.6%	1449

for the receiver part, it has a relatively low resource utilization that does not exceed 1.500 LUTs.

5.5 Evaluation on the AMD HACC Cloud Setup

We extend our evaluation to the HACC setup from AMD. While we do not run the attack itself, we evaluate the chances of uploading several malicious tenants to the same FPGA. Moreover, we evaluate the success of the covert channel communication between several tenants.

5.5.1 Success of having Multiple Malicious co-Tenants. HACC does not offer multi-tenancy. However, based on the recorded traffic of the cloud setup (see Section 4.2), we are able to emulate such scenario. In one location, HACC has 50 FPGAs. We record the traffic and instead of assigning each user an FPGA, we assume each five users will share an FPGA. We assume that the CSP will be clustering users on the same FPGA. It will try to fill an FPGA before using the other one.

The average usage time of a user during the recorded period was one hour and the maximum allowed is five hours. For each malicious tenant, we reserve the maximum period. Moreover, we assume that the malicious tenants will be uploaded separated by 15 minutes. We also assume that the attacker will flood a system with upload requests till the FPGAs are all full.

Figure 13 shows the success rate of the attack. We consider an attack is successful if the minimum number of tenants needed is uploaded to an FPGA while having at least one victim residing with them. The success rate gets lower with increasing the number of malicious tenants. If the attack needs two tenants to be successful,

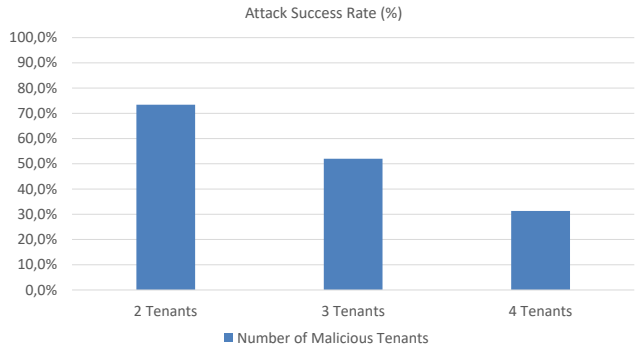

Figure 13: Attack success rate of coordinated malicious tenants on the Alveo U200 at the HACC cloud setup.

it has a 70% chance of success while for four tenants, the success chance of uploading the tenants together is around 30%. Note, that these results are highly dependent on the clustering policy of the CSP. If a different policy is used, e.g., considering the tenant's period of usage, or differently sized tenants are needed the results would change.

5.5.2 Success of Establishing Covert Communication. Next, we run the covert communication on the Alveo board using the setup shown in Fig. 8. We increase the number of communicating tenants from two to ten. For simplicity, we use the same type of attack for each of them. The communication significantly degrades when increasing the number of tenants communicating. This makes sense, as if only one of the tenants fails in the communication even once, the whole system fails because the slot assignment fails. Moreover, based on the type of the attack launching the communication the failures increase. For DES-based, the communications fail most significantly, while for latch-based the communication is more stable and successful.

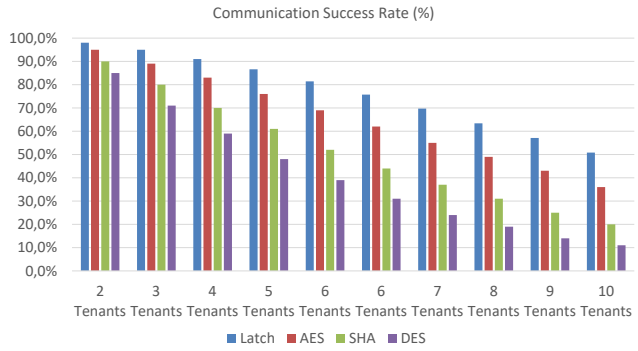

Figure 14: Communication success rate on the Alveo U200, with each extra tenant the ability to establish the communication drops significantly.

6 DISCUSSION

6.1 Distinction from State-of-the-art Single Tenant Attacks

A single-tenant attack is easy to detect [21, 41] by offline countermeasures and/or is easy to stop by online countermeasures [18, 19]. In contrast to them, we propose a stealthy attack using multiple small tenants composed of benign-looking circuits which are harder to detect by offline countermeasures. Furthermore, using multiple attackers makes locating and stopping them harder.

Our main contribution is that we show how multiple tenants can synchronize an attack. Our work shows that multiple small-sized tenants can coordinate to cause successful attacks. We show how this coordination can be done using a stealthy covert channel that evades detection.

An attack's success is directly related to resources used [19]. We show that reducing the tenant size does not stop the attack, as the attacker still can combine two attacks from two tenant regions each alone has a low rate of success will lead to a successful attack as the overall power consumption is higher now.

6.2 Calibration

The design of the TDCs required calibration. Deciding on the total chain length, initial delay value, and observable delay need manual iterations. Moreover, it changes from one FPGA generation to the other, e.g., going from Ultrascale to Ultrascale+. Therefore, we assume that the attacker will have access to an FPGA from at least the same generation as the one it attacks at hand. This assumption is reasonable, as cloud FPGAs are from the same families as the off-the-shelf development boards available for purchase.

6.3 Countermeasures

The main vulnerability that we exploit is the inability of online countermeasures to target multiple malicious tenants in parallel. In theory, LoopBreaker [19] can be extended to target several malicious tenants at the same time. To do so LoopBreaker needs multiple reconfiguration ports which is usually available on an FPGA [46]. However, the manufacturer allows that only one reconfiguration port is enabled at a time. Therefore, the manufacturer would need to allow having multiple reconfiguration ports working in parallel to enable the LoopBreaker extension.

Another countermeasure might be to interrupt the covert channel to prevent coordination, degrading the attacks to the level of single tenants as in subsection 5.3. To achieve that, existing countermeasures that work against side-channel attacks might be re-evaluated for our presented coordinated fault attacks [47, 48].

7 CONCLUSIONS

FPGAs are increasingly used to accelerate computation. To increase efficiency, one direction heavily discussed is multi-tenancy where several users benefit from the same FPGA instead of having lots of ideal resources. However, one of the main threats against multi-tenant FPGAs is power-hammering where a malicious tenant creates disturbances to the power distribution network of the FPGA. Countermeasures against power-hammering exist and make it infeasible for a single malicious tenant to perform a successful attack. In this work, we propose a method to realize a coordinated and synchronized attack from several tenants. We use a voltage-based covert channel to synchronize the attack and launch it successfully. Our coordinated attack, including its communication part, evades detection by state-of-the-art countermeasures.

REFERENCES

[1] M. G. Jordan, G. Korol, M. B. Rutzig, and A. C. S. Beck, "MUTECO: A Framework for Collaborative Allocation in CPU-FPGA Multi-tenant Environments", in *SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2021.

[2] J. González-Gómez, M. B. Sikal, H. Khdr, L. Bauer, and J. Henkel, "Smart detection of obfuscated thermal covert channel attacks in many-core processors", in *Design Automation Conference (DAC)*, 2023.

[3] S.-H. Lee, L. Wang, A. Khisti, and G. W. Wornell, "Covert communication with channel-state information at the transmitter", *IEEE Transactions on Information Forensics and Security (TIFS)*, 2018.

[4] J. González-Gómez, K. Cordero-Zuñiga, L. Bauer, and J. Henkel, "The first concept and real-world deployment of a gpu-based thermal covert channel: Attack and countermeasures", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023.

[5] K. Shahzad and X. Zhou, "Covert wireless communications under quasi-static fading with channel uncertainty", *IEEE Transactions on Information Forensics and Security (TIFS)*, 2021.

[6] J. Gonzalez-Gomez, L. Bauer, and J. Henkel, "Cache-based side-channel attack mitigation for many-core distributed systems via dynamic task migration", *IEEE Transactions on Information Forensics and Security (TIFS)*, 2023.

[7] L. Bossuet and C. A. Lara-Nino, "Advanced covert-channels in modern socs", in *IEEE Hardware Oriented Security and Trust (HOST)*, 2023, pp. 80–88.

[8] P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution", in *IEEE Symposium on Security and Privacy (S&P'19)*, 2019.

[9] L. Bauer, J. Becker, J. Henkel, F. Lesniak, and H. Nassar, "Adaptive application-specific invasive micro-architectures", in *Invasive Computing*. FAU University Pres, 2022.

[10] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "Survey and benchmarking of machine learning accelerators", in *High Performance Extreme Computing Conference (HPEC)*. IEEE, 2019, pp. 1–9.

[11] Z. Zhu, A. X. Liu, F. Zhang, and F. Chen, "FPGA Resource Pooling in Cloud Computing", *IEEE Transactions on Cloud Computing*, 2021.

[12] A. M. Caulfield, E. S. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J.-Y. Kim, D. Lo, T. Massengill, K. Ovtcharov, M. Papamichael, L. Woods, S. Lanka, D. Chiou, and D. Burger, "A cloud-scale acceleration architecture", in *International Symposium on Microarchitecture (MICRO)*, 2016.

[13] A. Putnam, A. M. Caulfield, E. S. Chung, D. Chiou, K. Constantinides, J. Demme, H. Esmaeilzadeh, J. Fowers, G. P. Gopal, J. Gray, M. Haselman, S. Hauck, S. Heil, A. Hormati, J.-Y. Kim, S. Lanka, J. Larus, E. Peterson, S. Pope, A. Smith, J. Thong, P. Y. Xiao, and D. Burger, "A reconfigurable fabric for accelerating large-scale datacenter services", in *International Symposium on Computer Architecuture (ISCA)*, 2014.

[14] Y. Zha and J. Li, "Virtualizing FPGAs in the cloud", in *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2020.

[15] T. La, K. Pham, J. Powell, and D. Koch, "Denial-of-Service on FPGA-based Cloud Infrastructures — Attack and Defense", *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2021, 2021.

[16] C. Bobda, J. M. Mbongue, P. Chow, M. Ewais, N. Tarafdar, J. C. Vega, K. Eguro, D. Koch, S. Handagala, M. Leeser *et al.*, "The future of FPGA acceleration in datacenters and the cloud", *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 15, no. 3, pp. 1–42, 2022.

[17] 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.

[18] G. Provelengios, D. Holcomb, and R. Tessier, "Mitigating voltage attacks in multi-tenant FPGAs", *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2021.

[19] H. Nassar, H. AlZughbi, D. Gnad, L. Bauer, M. Tahoori, and J. Henkel, "LoopBreaker: Disabling interconnects to mitigate voltage-based attacks in multi-tenant FPGAs", in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.

[20] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "Mitigating electrical-level attacks towards secure multi-tenant FPGAs in the cloud", *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2019.

[21] T. M. La, K. Matas, N. Grunchevski, K. D. Pham, and D. Koch, "FPGADefender: Malicious self-oscillator scanning for Xilinx UltraScale+ FPGAs", *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2020.

[22] M. M. Alam, S. Tajik, F. Ganji, M. Tehranipoor, and D. Forte, "RAM-Jam: Remote temperature and voltage fault attack on FPGAs using memory collisions", in *Fault Diagnosis and Tolerance in Cryptography (FDTC)*, 2019.

[23] G. Provelengios, D. Holcomb, and R. Tessier, "Power wasting circuits for cloud FPGA attacks", in *Field-Programmable Logic and Applications (FPL)*, 2020.

[24] D. R. Gnad, V. Meyers, N. M. Dang, F. Schellenberg, A. Moradi, and M. B. Tahoori, "Stealthy logic misuse for power analysis attacks in multi-tenant fpgas", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021, pp. 1012–1015.

[25] L. Alrahis, H. Nassar, J. Krautter, D. Gnad, L. Bauer, J. Henkel, and M. Tahoori, "MaliGNNoma: GNN-Based Malicious Circuit Classifier for Secure Cloud FPGAs", in *IEEE HOST*, 2024.

[26] G. Dessouky, A.-R. Sadeghi, and S. Zeitouni, "SoK: Secure FPGA Multi-Tenancy in the Cloud: Challenges and Opportunities", in *IEEE EuroS&P*, 2021.

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

[28] 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 *International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2018.

[29] M. Zhao and G. E. Suh, "Fpga-based remote power side-channel attacks", in *Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 229–244.

[30] J. Krautter, D. R. Gnad, and M. B. Tahoori, "Fpgahammer: Remote voltage fault attacks on shared fpgas, suitable for dfa on aes", *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, pp. 44–68, 2018.

[31] 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*, 2019.

[32] D. R. Gnad, J. Krautter, M. B. Tahoori, F. Schellenberg, and A. Moradi, "Remote electrical-level security threats to multi-tenant fpgas", *IEEE Design & Test*, 2020.

[33] J. Krautter, D. R. Gnad, and M. B. Tahoori, "Remote and stealthy fault attacks on virtualized fpgas", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021.

[34] A. Boutros, M. Hall, N. Papernot, and V. Betz, "Neighbors from hell: Voltage attacks against deep learning accelerators on multi-tenant fpgas", in *IEEE International Conference on Field-Programmable Technology (FPT)*, 2020.

[35] O. Glamočanin, L. Coulon, F. Regazzoni, and M. Stojilović, "Are cloud fpgas really vulnerable to power analysis attacks?" in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020.

[36] I. Giechaskiel, K. B. Rasmussen, and J. Szefer, "C^{3}apsule: Cross-fpga covert-channel attacks through power supply unit leakage", in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020.

[37] I. Giechaskiel, K. Rasmussen, and J. Szefer, "Reading between the dies: Cross-slr covert channels on multi-tenant cloud fpgas", in *International Conference on Computer Design (ICCD)*. IEEE, 2019.

[38] D. R. E. Gnad, C. D. K. Nguyen, S. H. Gillani, and M. B. Tahoori, "Voltage-based covert channels using FPGAs", *ACM Transactions on Design Automation of Electronic Systems (TODAES)*), 2021.

[39] M. Gross, R. Kunzelmann, and G. Sigl, "CPU to FPGA power covert channel in FPGA-SoCs", Cryptology ePrint Archive, 2023.

[40] S. Tian and J. Szefer, "Temporal thermal covert channels in cloud fpgas", in *International Symposium on Field-Programmable Gate Arrays (FPGA)*. ACM/SIGDA, 2019.

[41] J. Chaudhuri and K. Chakrabarty, "Diagnosis of malicious bitstreams in cloud computing FPGAs", *Transactions on Computer Aided Design of Integrated Circuits & Systems (TCAD)*, 2023.

[42] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits", in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 1989.

[43] P. Jamieson, T. Becker, P. Y. K. Cheung, W. Luk, T. Rissa, and T. Pitkänen, "Benchmarking and evaluating reconfigurable architectures targeting the mobile domain", *ACM Transactions on Design Automation of Electronic Systems (TODAES)*), 2010.

[44] J. Pistorius, M. Hutton, A. Mishchenko, and R. Brayton, "Benchmarking method and designs targeting logic synthesis for fpgas", in *International Workshop for Logic Synthesis (IWLS)*, 2007.

[45] "Heterogeneous Accelerated Compute Clusters (HACC)". [Online]. Available: https://www.amd-haccs.io/index.html

[46] *UltraScale Architecture Configuration (v1.18) UG570*, Xilinx, Inc., 2023.

[47] J. Krautter, D. R. Gnad, F. Schellenberg, A. Moradi, and M. B. Tahoori, "Active fences against voltage-based side channels in multi-tenant fpgas", in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–8.

[48] O. Glamočanin, A. Kostić, S. Kostić, and M. Stojilović, "Active wire fences for multitenant fpgas", in *International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*. IEEE, 2023, pp. 13–20.