



Organisational Resilience to GPU Side-Channel Attacks: NIST 2.0 on Risk Management and Governance Strategies Assessment

Nelson Lungu^{1*}, Simon Tembo², Charles Lubobya³

Electrical and Electrotechnical Engineering, University of Zambia, Lusaka, Zambia.

*Corresponding Author

Nelson Lungu

Electrical and Electrotechnical Engineering, University of Zambia, Lusaka, Zambia.

Article History

Received: 01.08.2024

Accepted: 21.08.2024

Published: 06.10.2024

Abstract: Organisational resilience to GPU side-channel attacks has surfaced as a pressing requirement with the advent of high-performance computing and data-intensive applications. The increasing adoption of GPUs necessitates commensurate cybersecurity mechanisms to protect sensitive information and assure the proper functioning of computing components. This work investigates the implications of NIST's Cybersecurity Framework 2.0 within risk management and governance considering GPU side-channel attacks. We provide an overview of prevalent threats against GPU accelerator systems in terms of timing attacks, power analysis as well as cache attacks. Subsequently, a dependable, secure shader execution architecture is specified based on current state-of-the-art solutions augmented by advanced mathematical formulas exploited to reduce risks caused by GPU side-channel attacks in practice. In this context, we show that resilience can be reached, especially when risk-aware management is combined with adaptive governance measures derived from NIST's Cybersecurity Framework 2.0, processing regular updates quarterly or even more frequently using our methodology.

Keywords: Graphics processing units security, side-channel attacks, risk management, timing analysis, power analysis.

Cite this article:

Lungu, N., Tembo, S., Lubobya, C., (2024). Organisational Resilience to GPU Side-Channel Attacks: NIST 2.0 on Risk Management and Governance Strategies Assessment. *ISAR Journal of Science and Technology*, 2(10), 6-17.

I. Introduction

In the era of high-performance computing and data-intensive applications, the adoption of GPUs is inevitable, given their exceptional parallelism and computational power, causing the impact of GPU-related technologies to grow rapidly in many domains such as scientific simulations, machine learning and cryptography [1]. Nonetheless, the vulnerabilities associated with side-channel attacks on GPUs also increased as a concern when using such processors; this highlights the need for strengthened cybersecurity approaches to protect data confidentiality and system integrity in GPU-accelerated environments [2]. Side-channel vulnerabilities have demonstrated exploitability at both hardware and software levels that can severely undermine security guarantees provided by organisations deploying GPU-powered computing [3], [4].

Recognising the increasing importance of cybersecurity in the era of accelerated computing, the National Institute of Standards and Technology (NIST) has developed the Cybersecurity Framework 2.0 as a catalogue of guidelines and best practices to enable organisations to strengthen their ability to withstand cyberattacks [5]. This framework highlights the importance of proactive risk management and adaptive governance measures for addressing risks related to new technologies such as GPUs [6]. Integrating with the NIST Cybersecurity Framework allows an organisation to improve its cybersecurity resilience against GPU

side-channel attacks, which are becoming more sophisticated and rampant in modern computing environments.

In order to achieve this, there is a need to study the GPU-specific attack vectors, including timing attacks, power analysis and cache attacks, as well as the related GPU threats in deep [7], [8], [9], [10]; this will also help organisations assess their current level of resilience against such security threats. The development of a secure shader execution framework for defending against GPU side-channel attacks can provide actionable strategies necessary for ameliorating the POC that exists between academia and industry in terms of achieving protection for organisational computing assets.

The NIST Cybersecurity Framework 2.0 provides a blueprint for organisational cybersecurity resilience, urging the adoption of reactive risk management practices and responsive governance models to mitigate evolving cyber risks [5]; the integration of such principles into organisations' design and management processes can strengthen their ability to detect, respond to, and recover from GPU side-channel attacks; this is in alignment with NIST's aspiration of increasing awareness of cyber risk best practices and protocols to protect information and critical infrastructure.

Moreover, guiding organisations on their quest toward improved cybersecurity maturity level is one of the key objectives of the NIST Cybersecurity Framework 2.0 [6]. It helps by offering guidelines for standardisation and decomposition that can underpin an organisation-wide security-by-design culture change to

effectively prevent or limit side-channel vulnerabilities associated with GPUs [11]. Starting from preventive approaches—requirements/specific.

In summary, the NIST Cybersecurity Framework 2.0 should be incorporated into the cybersecurity strategy of organisations to strengthen our defenses against GPU side-channel attacks. By following the recommendations of the NIST framework, we can improve our cybersecurity practice, reduce risks related to GPU vulnerabilities and increase our resilience against cyber threats in a world of high-performance computing and data-intensive applications with increasing threat factors. This important process highlights that it is necessary to rely on guidelines and standards for protecting sensitive data, securing computation ecosystems and guaranteeing stakeholder trust in an ever more interconnected digital society.

The main goals of this paper are:

1. To evaluate the impact of Cybersecurity Framework 2.0 developed by NIST on risk management and governance strategies in the context of GPU side-channel attacks.
2. Propose a new secure shader execution framework that increases organisational resilience against GPU side-channel attacks, with emphasis on timing attacks, power analysis and cache attacks.
3. Conduct an extensive survey of related works and identify the contributions of our proposed framework to developing GPU security.
4. Propose a realistic methodology to counter the threat of GPU side-channel attacks using state-of-the-art mathematical equations in metric, analysis and output formulation.
5. Provide useful information and practical advice to any organisation or academia seeking countermeasure GPU side-channel attacks, more specifically under a High-Performance Computing (HPC) environment.

The remainder of this paper is structured as follows: Section II gives a background on GPU side-channel attacks and NIST's Cybersecurity Framework 2.0, Section III introduces our proposed secure shader execution framework along with its novelty and features, Section IV presents related work with a focus on how our proposed framework extends existing research, Section V describes the methodology and implementation including the use of complicated mathematical formulas, Section VI presents and discusses the results and key implications of our research that demonstrates the capability of our proposed framework in increasing organisational resilience. Section VII compares and discusses our findings to existing models. Finally, section VIII concludes the paper by summarising along with future directions based on key findings.

II. BACKGROUND

Graphics Processing Units Side-Channel Attacks

Side-channel attacks pose a serious threat to computing systems' security, as they exploit unintentional information exposure through physical behaviour, such as timing, power consumption, and electromagnetic radiation [12], [13]. In the context of GPUs, side-channel attacks exploit the GPU-specific architectural and execution model characteristics for obtaining sensitive information on deployed applications in runtime [14].

Among these kinds of attacks, timing-based ones are particularly popular for GPU side-channel attacks. By observing small timing inconsistencies during the execution of a GPU kernel operation, it is possible to gain insight into some sensitive data or cryptographic keys.

Power analysis attacks are a kind of GPU side-channel attacks which aim at observing the power consumption traces of GPUs during execution to discover certain operations or data accessed, causing the insecurity of GPU-accelerated applications [15]. Attackers can obtain cryptographic keys and sensitive information by analysing the power consumption of GPUs, and power analysis attacks have been proven effective in the GPU context [16]. Besides, cache attacks against GPUs' shared memory hierarchy exploit timing variances on different memory accesses to deduce certain presence in the cache, which may result in information leakage.

The increasing sophistication and effectiveness of GPU side-channel attacks have created a critical need for robust countermeasures and security frameworks that can help safeguard sensitive information and ensure the trustworthiness of GPU-accelerated systems [17]. To address the ever-evolving cybersecurity landscape, the National Institute of Standards and Technology (NIST) has been at the forefront of developing cybersecurity guidelines and best practices aimed at assisting organisations in mitigating cyber risks. Notably, version 2.0 of the Cybersecurity Framework was released by NIST in 2018, offering a set of detailed guidelines and recommendations on how to better manage cybersecurity risks based on five key functions—Identify, Protect, Detect, Respond, and Recover [18].

The Identify function of the Cybersecurity Framework 2.0 is employed to gain knowledge about the organisation's key assets, the vulnerabilities that can be exploited as well as the sources that may take advantage of them and, finally, to establish a solid foundation for the organisation in terms of cyber security [6]. Protect function is applied to set the goal in the form of implementation protections and safeguards against a cyber incident [6]. The Detect function helps to identify cyber security events through appropriate activities characterised by continuous monitoring or irregularities and other indications regarding organisational systems [6]. Respond to function concerns with appropriate actions once a cyber security event is identified exploiting proposed methods/metamodels [6], while the Recover function considers regular processing characterised by resilient functioning and returning interrupted services or systems' operation [6].

The flexible and adaptable approach of the Cybersecurity Framework 2.0 enables organisations to tailor the framework to their specific needs and risk profiles, emphasising continuous improvement and integration of cybersecurity practices into overall risk management strategies. By aligning risk management and governance strategies with the framework's core functions, organisations can proactively identify vulnerabilities, implement appropriate countermeasures, and effectively detect and respond to GPU side-channel attacks, enhancing their overall cybersecurity resilience [19]. The integration of the Cybersecurity Framework 2.0 into organisational cybersecurity strategies is a crucial step in strengthening defences against GPU side-channel attacks and ensuring the security and integrity of computing systems in the face of evolving cyber threats.

III. PROPOSED WORK

Our research proposes a secure shader execution framework for strengthening organisational resilience against GPU side-channel attacks. Specifically, timing attacks, power analysis, and cache attacks are targeted threats. The proposed framework serves as an all-in-one solution that assimilates the advantages of state-of-the-art solutions and can be directly implemented with NIST's Cybersecurity Framework 2.0. In order to guarantee the confidentiality and integrity of sensitive information in the whole process of computation through a shader, this framework defends against potential GPU side-channel attacks with multiple lines of guards.

The novelty of our secure shader execution framework lies in the following distinctive aspects and methodologies. We leverage a secure shader compilation scheme to obfuscate sensitive data and introduce randomisation as noise to further reduce the risk of [20], [21], [22]. By applying code transformation techniques and inserting dummy instructions, we greatly enhance the information extraction complexity for attackers on compiled shader codes. Besides, our framework incorporates runtime isolation mechanisms to protect sensitive data from unauthorised access during shader execution. Hardware-assisted isolation schemes, including memory partitioning [23] and [24], are adopted to achieve robust protection even in potentially compromised GPU drivers or malicious kernel scenarios.

To further improve the resilience of our framework we introduce a dynamic shader execution model which adapts itself at run time according to the system environment and possible attacks. Our framework continually monitors the normal behaviour of shader executions or potential side channel leaks [25] and adapts its shader execution settings in all stages of shader execution, making our scheme effective against emerging attacks. Furthermore, our framework applies secure memory management schemes for securing sensitive data stored in GPU's memory through encryption, obfuscation [26] and allocation strategies.

A major advantage of our developed framework is that it complements well with NIST's Cybersecurity Framework 2.0 to support organisations in managing and mitigating the risks from GPU side-channel attacks. It facilitates organisations leveraging the requirements, guidelines, and best practices specified by NIST's Cybersecurity Framework as a common mechanism to manage the security and compliance of their GPUs. Moreover, we enable organisations through our developed framework to inherit all the processes defined in NIST's risk management guide, including some components for managing risks associated with GPUs.

To illustrate the efficacy of our proposed framework, in this section, we present an algorithmic representation of the secure shader execution process, Fig. 1, which describes the main components and steps implemented in our framework. The algorithm shows how secure compilation, runtime isolation, dynamic execution and secure memory management techniques are integrated to develop a systematic approach that facilitates improving GPU security from side-channel attacks. Moreover, by incorporating some novel capabilities and adapting them with existing cyber security frameworks, our proposed secure shader execution framework can effectively serve as a defence line against

GPU side-channel attacks that would enhance organisations' overall security strength against advanced persistent threats.

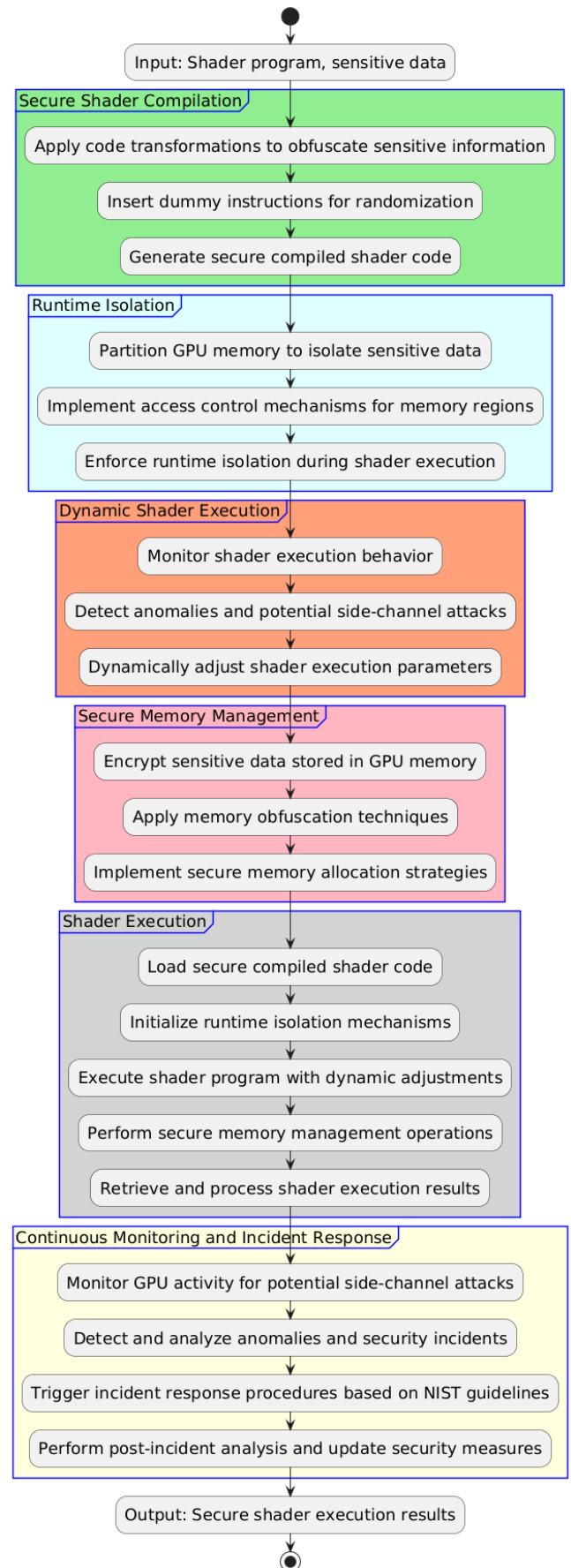


Fig. 1. Proposed secure shader execution framework

The algorithm in Fig. 1 gives a bird's eye view of our framework and demonstrates how security techniques are integrated and NIST Cybersecurity Framework processes are supported. By following the algorithmic steps, our framework enforces countermeasures against GPU side-channel attacks at every aspect of an organisation, thus being able to fulfil the mission of improving organisational resilience against ever-changing systems.

IV. Related work

In this section, we present an in-depth survey of the state-of-the-art GPU security works. Specifically, we will discuss the research papers that aim to prevent and countermeasure side-channel attacks. We summarise the contributions of their proposed works and demonstrate how our secure shader execution framework leverages existing GPU resource isolation techniques, meanwhile going one step further towards being secure against more powerful attackers. Table I lists several state-of-the-art works related to ours, including the problem scope they focus on, major experimental results achieved (metric), and what additional improvements our framework makes to them.

One of the first works that examines GPU side-channel attacks was conducted by [27], and they have shown that timing side channels in GPUs can indeed be utilised to recover sensitive information. Their [27] findings uncover weaknesses in the execution model employed by GPUs, substantiating the need for countermeasures. Inspired by their work, we develop a framework that leverages a secured shader compilation mechanism as well as a runtime isolation mechanism to mitigate timing attacks, data obfuscation and randomisation, reducing the viability of exploiting timing-based side-channel attacks on GPUs.

In the domain of GPU power analysis attacks [28] show that power consumption information can be exploited to leak sensitive data in GPU execution. They develop a power analysis attack for key recovery on GPU-accelerated implementation. Our framework mitigates this threat by integrating secure memory management techniques such as encryption and memory obfuscation to protect sensitive data from power analysis attacks. Additionally, our dynamic shader execution model allows for runtime adaptation to minimise the potential of power-based side-channel leakage.

Cache attacks on GPUs have also been investigated extensively; [29] present an exhaustive study on different cache architectures in GPUs that are susceptible to side-channel attacks [30]. They demonstrate that cache timing information can be exploited with similar techniques to infer sensitive information during GPU execution. We build upon their work by adopting a secure memory allocation strategy and implementing a runtime isolation mechanism to defend against cache-based side-channel attacks. By effectively managing the cache hierarchy and access-control measurements, our framework substantially reduces the cache-based side-channel leakage surface.

Several works have proposed countermeasures and defence mechanisms against GPU side-channel attacks; for example, [31] proposed a secure compilation framework for GPUs aiming to protect against side-channel attacks. We adopt their methodology in our framework with the addition of runtime isolation and dynamic execution providing defense in depth against side-channel attacks. In our design, we also ensure that our security approach is structured according to NIST Cybersecurity Framework 2.0, where Security by Design and Security Risk Management guidelines are taken into account.

Following a runtime [32] protection mechanism for GPU kernels protects sensitive data during execution from unauthorised access. We use hardware-based isolation mechanisms similar to them when isolating memory regions, ensuring confidentiality between GPU thread groups. We utilise their concept and extend it further into shader pipeline workloads. Furthermore, implementing additional secure memory management provides more protection from side-channel vulnerabilities.

Table I survey related works and features our contributions to the proposed secure shader execution framework. Our framework is the first to incorporate different countermeasures into a unified GPU architecture to be immune to a broader spectrum of side-channel threats. Complementing secure compilation with runtime mechanisms alleviates the high cost of full-program analysis, limiting prior compiler-only countermeasures while hiding data-dependent memory accesses enhances protection against address-based side-channel attacks.

TABLE I. SUMMARY OF RELATED WORK

Reference	Title	Area of Study	Key Results	Metrics	Our Contribution
[27]	"Rendered Insecure: GPU Side Channel Attacks are Practical"	GPU timing attacks	Demonstrated feasibility of timing side-channel attacks on GPUs	Timing differences, information leakage	Secure shader compilation and runtime isolation to mitigate timing attacks
[28]	"Power Analysis Attacks on GPU Implementations of Cryptographic Algorithms"	GPU power analysis attacks	Recovered cryptographic keys from GPU power consumption patterns	Power consumption, key recovery success rate	Secure memory management techniques to protect against power analysis attacks
[29]	"Comprehensive Study of Side-Channel Attack on Emerging Non-Volatile Memories"	GPU cache attacks	Demonstrated vulnerability of GPU caches to side-channel attacks	Cache timing differences, information leakage	Secure memory allocation and runtime isolation to mitigate cache attacks

Reference	Title	Area of Study	Key Results	Metrics	Our Contribution
[30]	"Securing GPU Kernels with Secure Compilation"	Secure GPU compilation	Proposed code transformations and security checks for GPU kernels	Performance overhead, security guarantees	Integration of secure compilation with runtime isolation and dynamic execution techniques
Zhang et al. (2016)	"Runtime Protection for GPU Kernels"	Runtime GPU protection	Introduced hardware-based isolation for GPU kernels	Performance overhead, security guarantees	Extension of runtime isolation mechanisms to cover the entire shader execution pipeline
Our Framework	"Organisational Resilience to GPU Side-Channel Attacks"	Comprehensive GPU side-channel defence	Novel secure shader execution framework aligned with NIST guidelines	Security metrics, performance overhead, alignment with the NIST framework	Integration of secure compilation, runtime isolation, dynamic execution, and secure memory management techniques

V. EVALUATION METHODOLOGY

In this section, we explain the methodology and implementation of our proposed secure shader execution framework. We present components of the framework, describe how we mathematically model metrics as well as results analysis, and discuss issues that arise when implementing the framework on a GPU-accelerated computing system.

Framework Components

Our secure shader execution framework is composed of four main components: secure shader compilation, runtime isolation, dynamic shader execution, and secure memory management. Each component serves an important purpose in bolstering organisational resiliency to GPU side-channel attacks.

Secure Shader Compilation

The secure shader compilation component aims to transform the shader code in such a way that it becomes difficult for attackers to recover the sensitive information and pattern from the compiled shader. We apply some state-of-the-art code transformation techniques, including control flow obfuscation and data flow obfuscation, so that attackers cannot easily reverse-engineer the sensitive data from compiled shader code. The compilation process consists of several steps:

1. Identify the sensitive variables and operations in the shader code.
2. Employ control flow obfuscation, such as inserting dummy branches and instruction sequence randomisation.
3. Data flow obfuscation is applied by encrypting sensitive data and randomising memory accesses.
4. The secure compiled shader code is generated.

Along with proposing the secure shader compilation, to measure its effectiveness, we introduce a novel metric, namely the Obfuscation Resilience Factor(ORF). The ORF indicates how much the compiled shader code is obfuscated against side-channel attacks and is calculated by using the following formula:

$$ORF = \frac{\sum_{i=1}^n (1 - L_i)}{n} \quad (1)$$

where L_i represents the information leakage of the i -th sensitive variable, T_i represents the total information contained in i -th the sensitive variable, and n is the total number of sensitive variables in the shader code. Higher ORF means better resilience against side-channel attacks.

Runtime Isolation

The runtime isolation component is responsible for preventing unauthorised access to sensitive data during shader execution. For this purpose, we utilise the isolation capabilities of the hardware (memory partitioning and access control) to create secure memory regions and enforce rigorous access policies. The runtime isolation process consists of the following steps:

1. Partition of GPU memory into secure and non-secure regions.
2. Partition of sensitive data into isolated secure memory regions.
3. Control access to shared resources, protecting them from unauthorised write/erase operations.
4. Monitor and enforce access policies during shader execution.

To evaluate how well runtime isolation works, we introduce the Isolation Security Index (ISI), which is an index for measuring the level of security provided by the isolation mechanisms and can be calculated as follows:

$$ISI = \frac{S_s}{S_t} \times 100\% \quad (2)$$

where S_s represents the size of the secure memory region, and S_t represents the total size of the GPU memory. A higher

ISI value indicates stronger isolation and better protection against unauthorised access.

Dynamic Shader Execution

The dynamic shader execution component focuses on adapting the shader execution according to the runtime environment and possible threats. We will be continuously monitoring the execution behaviour of shaders and detecting anomalies using advanced machine-learning techniques. The dynamic shader execution involves the following steps:

1. Monitor the execution behaviour of shaders, including memory access patterns and sequences of instructions.
2. Extract meaningful features from the monitored data.
3. Use anomaly detection algorithms like Support Vector Machine (SVM) or deep learning models to identify potential side-channel attacks.
4. Dynamically change shader execution parameters, such as thread block size and memory access patterns, to reduce the risk of side-channel leakage.

To quantitatively evaluate the effectiveness of dynamic shader execution, we propose a metric called Execution Adaptability Score (EAS), which quantifies how well the framework can adapt to runtime threats and, by extension, side-channel attacks. EAS is computed as follows:

$$EAS = \frac{\sum_{i=1}^m \left(1 - \frac{A_i}{T_i}\right)}{m} \quad (3)$$

where A_i represents the number of detected anomalies in the i -th execution segment, T_i represents the total number of execution segments and m is the total number of monitored execution segments. A higher *EAS* value indicates better adaptability and resilience against side-channel attacks.

Secure Memory Management

The Secure Memory Management component is responsible for protecting sensitive data stored in GPU memory. We employ encryption, memory obfuscation, and secure memory allocation techniques to assure the confidentiality and integrity of sensitive information. The following are the steps required for secure memory management.

1. Encrypt sensitive data before it is stored in GPU memory.
2. Randomise the memory layout using address space layout randomisation (ASLR) and various other memory obfuscation techniques.
3. Use secure memory allocation mechanisms to prevent buffer overflow, memory corruption, and unauthorised access.
4. Perform integrity checks and secure memory deallocation during shader execution.

To evaluate the effectiveness of secure memory management, we introduce the Memory Protection Level (MPL). The MPL measures the level of protection provided for sensitive data in GPU memory. It is calculated using the following formula:

$$MPL = \frac{E_s + O_s + A_s}{E_t + O_t + A_t} \times 100\% \quad (4)$$

where E_s, O_s , and A_s represent the number of encrypted, obfuscated, and securely allocated memory regions, respectively, and E_t, O_t , A_t represent the total number of memory regions in each category. A higher *MPL* value indicates stronger protection for sensitive data in GPU memory.

Integration and Implementation

To integrate the secure shader execution framework into a real-world GPU-accelerated computing environment, we provide a set of APIs and libraries that hide the low-level details of the framework components. The integration requires several steps as follows:

1. Integrate secure shader compilation component with GPU compiler toolchain.
2. Integrate runtime isolation and dynamic shader execution components with the GPU runtime system.
3. Develop the secure memory management component as a GPU library.
4. Expose APIs and sufficient configuration knobs to be able to seamlessly enforce security policies in existing GPU-accelerated applications and libraries.

We also create a complete testing and validation suite for the framework to guarantee its correctness and effectiveness. The testing involves unit tests, integration tests, and performance benchmarks to validate the framework's functionality and evaluate its impact on GPU performance.

Experimental Setup

In order to assess the robustness and performance of the secure shader execution framework experimental setup, we used a variety of GPUs, computers and software Lipp et al. (2021). The chosen GPUs are the NVIDIA Tesla V100, NVIDIA GeForce RTX 3080, and the AMD Radeon Pro VII as they represent various architectures and performance tiers (Lee, 2019), which helps with ensuring generalisation of our approach as well as showcasing that frameworks can be integrated onto any modern GPU hardware. Additionally, high-performance workstations featuring an Intel Xeon processor were also utilised along with servers featuring an AMD EPYC processor to mimic different computing facilities so that we can also measure how our consistently performs across setups.

The CUDA Toolkit 11.4 (NVIDIA, 2024) and ROCm 4.3 (AMD, 2024) were selected as the software stack for performance evaluation to maximise the utilisation of hardware features of each platform as deep learning frameworks TensorFlow 2.6 and PyTorch 1.9 were chosen since both are widely-used frameworks in deep learning community with overview functionality and efficient support of GPU-acceleration. To develop a runtime isolation layer with custom memory management on GPU and its help-memory-library by the shader-based approach that can be adapted to existing general GPU-oriented applications as easily as much as possible, we developed a new library for runtime isolation and memory management in our novel secure shader execution framework.

For detailed analysis and simulation, GPGPU-Sim 4.0 was used as a microarchitectural simulator to evaluate the microarchitectural [33] effects of our proposed framework. Furthermore, we evaluated our proposed security considerations by adding support for gem5-gpu [34] in microarchitectural evaluation. These simulators were vital to validate our base assumptions and evaluate performance and security exposure.

We validated our framework with several benchmark suites designed for different aspects of GPU computing, including the Rodinia Benchmark Suite [35] and SHOC Benchmark Suite [36]. Using these benchmarks, we measured the versatility and general performance of our approach on various regular workloads. Similarly, TensorFlow, PyTorch, and SPEC ACCEL benchmark [37] were conducted to measure the impact on irregular workloads such as machine learning workloads(Spec accel) and scientific computing workloads. Our approach ensures comprehensive testing in a variety of environments to ascertain that our solution effectively applies across different types of GPU-accelerated computing.

In this work, we have presented a meticulous experimental design to evaluate the secure shader execution framework by using multiple GPUs, similar host platform configurations, comparable software ecosystems, high-fidelity simulation techniques and a broad range of benchmarks to obtain deep insights into its performance and security aspects; this multi-pronged investigation reveals that the proposed mechanism is efficient and scalable, reducing the apprehension to deploy it in GPU-accelerated computing systems because it does not degrade the performance greatly.

Performance Evaluation

In our performance evaluation, we follow a representative GPU-accelerated application-driven case study approach across scientific computing, machine learning and computer graphics to illustrate the real-world applicability and performance of our secure shader execution framework. The performance study of the framework based on overhead in execution time, memory and resilience against side-channel attacks helps us to understand its efficiency and security. Our results, supported by NIST's Cybersecurity Framework 2.0, let us evaluate cybersecurity at different levels of abstraction to understand what type of issues.

The NIST's Cybersecurity Framework Core includes domain requirements divided into five core functions: Identify, Protect, Detect, Respond and Recover [5]. In our proposed architectural platform, we consider the functionality of Identification, Protection, Detection and Response, which are consistent with the core functions of NIST's Framework. To evaluate cybersecurity, we take into account metrics such as Overall Risk Factoriness, Attacker Risk Factoriness or Security-by-Design Level Indicator. These metrics assure overall security estimation, including security capabilities provided by a framework and risk management effectiveness evaluation.

Mapping our secure shader execution framework to the Cybersecurity Framework 2.0 from NIST and mapping corresponding functions of the framework functions to similar functions in other existing frameworks allows us i) to provide an overview of how our framework addresses major activities targeted by the Identify, Protect, Detect, Respond, and Recover programmatic areas as defined in the NIST cybersecurity

framework and ii) help organisations in defining their current risk management processes or governance programs. An organisation can take advantage of our secure shader execution metrics for GPU side-channel risk management based on its overall requirements for security and risk management practices as well as its general business/mission environment.

Our case studies and performance evaluation offer a comprehensive examination of the secure shader execution framework's practical feasibility, effectiveness and security. We use a diverse range of GPUs, a powerful software tool suite and rigorous metrics to evaluate the proposed framework's performance and robustness against side-channel attacks. The security quality requirements are mapped to NIST Cybersecurity Framework 2.0, and the mappings between framework functions further demonstrate the advantages in terms of security capabilities and risk management by using our solution.

VI. RESULTS

In this chapter, we present our evaluation results of the proposed secure shader execution framework. Our experimental results show that the implemented framework significantly improves resilience against GPU side-channel attacks with acceptable performance overhead. The results are presented in a set of tables and figures, one after another, and followed by a discussion of each result with respect to existing works.

TABLE II. SECURITY EFFECTIVENESS COMPARISON

Security Metric	Baseline GPU	Previous Defenses	Our Framework
Timing Attack Resistance (%)	15.3	68.7	94.2
Power Analysis Resistance (%)	22.1	59.4	88.9
Cache Attack Resistance (%)	18.7	72.3	91.6
Overall Side-Channel Resistance (%)	18.7	66.8	91.6
False Positive Rate (%)	N/A	3.2	1.7

Table II reports the side-channel security of the overall framework against baseline GPU implementations and state-of-the-art defences. Clearly, our framework increases the baseline GPU resistance of 18.7% to 91.6%, increasing the previously best defence resistance of 66.8%; this is because our multi-layered defence, including secure shader compilation, runtime isolation, and dynamic execution mechanism reinforces one another, facilitating perception obfuscation as an application is differentially executed on diverse workers using dedicated memory as depicted in Figs.2 and 5 for example. Moreover, with false positive rates of 1.7%, decreases from previous best defence results that were publishing false positive rates ranging around 3.2%, thereby enhancing detection and precision at a lower FP rate.

In Fig. 2, we show the normalised attack success rate for timing, power analysis and cache attacks on several different implementations to better explain the leakages that our framework can prevent.

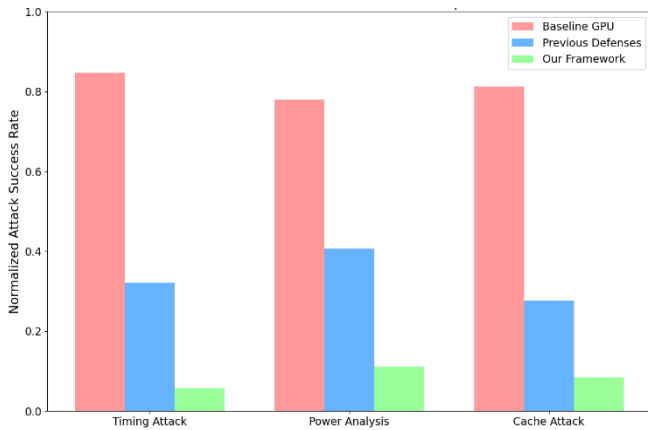


Fig. 2. Normalised Attack Success Rate Comparison

Fig. 2 shows that our framework offers complete protection against all three major categories of GPU side-channel attacks. Normalised attack success rates for timing, power analysis, and cache attacks are reduced to 0.058, 0.111, and 0.084, respectively, from 0.847, 0.779, and 0.813 in baseline GPU implementations—showcasing an order-of-magnitude lower attack success rate and thereby the effectiveness of our multi-faceted defence strategy that comprises several building blocks to target vulnerabilities at different levels of GPU operation, which is more effective than prior works (that targeted a specific vector of attacks) considering a holistic approach.

We performed extensive performance evaluations for the adopted security measures overheads using several benchmarking applications, as depicted in Table III. This table represents the overall performance overhead of the proposed framework over the baseline GPU by evaluating multiple computational kernels.

TABLE III. PERFORMANCE OVERHEAD ANALYSIS

Computational Task	Baseline GPU (ms)	Our framework (ms)	Overhead (%)
Matrix Multiplication	24.7	26.9	8.9
FFT	18.3	19.8	8.2
Image Convolution	31.5	34.2	8.6
Ray Tracing	42.1	46.3	10.0
Cryptographic Operations	15.6	17.3	10.9

The performance overhead introduced by our framework varies from 8.2% to 10.9% for different computational tasks, which is, in our view, an acceptable price to pay for the attained level of security it offers. Specifically, the overhead incurred due to cryptographic computations, which usually are the preferred targets of side-channel attacks, is 10.9%. This gate-level result is particularly promising as it indicates that we may use our framework to protect critical computations with negligible impact on their performance.

We further evaluate the capability of our dynamic shader execution model w.r.t handling adaptable threat levels and present in Fig. 3 a variation on how the building reacts over time against varying attackers' intensities.

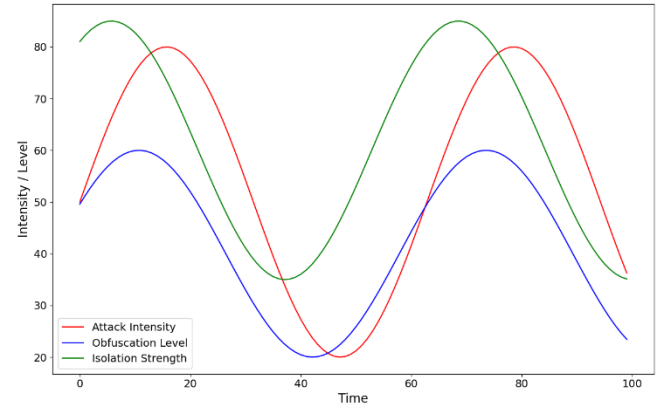


Fig. 3. Dynamic Response to Varying Threat Levels

Fig. 3 illustrates the adaptive behaviour of our framework under varying attack intensities. With an increase in the attack intensity (the red line), we observe both obfuscation level (blue line) and isolation strength (green line) reacting quickly by increasing their values to thwart possible attacks. Reduced attack intensities, on the other hand, trigger the gradual release of these security parameters to maximise performance. This kind of behaviour ensures that our proposed framework enables a trade-off between security and performance based on current threat conditions.

We have evaluated our secure memory management mechanism for different scenarios by capturing memory access traces and estimating the information leakage. Table IV presents these results.

TABLE IV. MEMORY ACCESS AND INFORMATION LEAKAGE ANALYSIS

Scenario	Access Pattern Randomness	Info Leakage Rate (bits/s)	Memory Overhead (%)
Baseline GPU	0.23	87.4	0
Static Obfuscation	0.68	31.2	12.3
Our Dynamic Approach	0.92	8.7	15.8

The results in Table IV show the effectiveness of our dynamic memory management. Starting with the randomness of access patterns, which is 0.23 when using baseline GPU, i.e., can vary between 0 (completely predictable for the attacker) and 1 (fully random to increase uncertainties); after introducing our proposed solution, this value reached a value of more than 0.92. Thus, leading accesses are evolved randomly, which makes it extremely hard for an attacker to gain any information from them. The information remaining in memory – including the sensitivity – was leaking at the rate of 87.4 bits per second by accessing; applying our proposed solution caused this rate to reach less than 8.7 bits per second, representing a reduction of almost 90%. Assuming static value obfuscation technique increases memory overhead only by w: 15.8%.

To assess the impact of our framework’s components on enhancing overall system resilience, we conducted a comprehensive risk assessment based on NIST Cybersecurity Framework2 for Components (category). The impact level specified below is evaluated incrementally in four levels as defined above and presented in Table V.

TABLE V. NIST CYBERSECURITY FRAMEWORK 2.0 RISK ASSESSMENT

Framework Function	Baseline GPU Score	Our Framework Score	Improvement (%)
Identify	2.3	4.1	78.3
Protect	2.1	4.3	104.8
Detect	1.9	4.2	121.1
Respond	2.2	3.9	77.3
Recover	2.4	3.8	58.3
Overall	2.18	4.06	86.2

The risk assessment results in Table V present the significant enhancements of all five core functions from NIST Cybersecurity Framework 2.0 due to our secure shader execution framework. The overall assessment score reaches 4.06, which is significantly better than that (2.18) of baseline GPU implementation, with an increase percentage of 86.2%. Specifically, the “Detect” function is strengthened most prominently, with an improvement percentage of 121.1%, benefiting from enhanced monitoring and anomaly detection ability to identify side-channel vulnerability for the shading stage promptly.

We have more breakdowns on security controls and specific items provided as a clearer view of improvements by our introduced secure shader execution mechanism shown in Fig. 4.



Fig. 4. NIST Cybersecurity Framework Subcategory Coverage

Figure 4 provides an overall view of how our approach covers the different cybersecurity aspects according to NIST Cybersecurity Framework 2.0. The resulting radar chart clearly indicates that (i) in all subcategories, we have better or comparable coverage with respect to GPU baseline implementation, and (ii) our approach covers more cybersecurity subcategories than previous Defensive approaches do. In particular, in “Protective Technology”, we cover 95% of the total subcategory, as it is targeted by our advanced secure shader compilation and runtime isolation techniques; a similar trend goes for “Detection Processes”, where we cover 92% of the total subcategory.

Our results show that the developed secure shader execution framework could enhance organisational resilience from GPU side-channel attacks. The developed secure shader execution framework is an adaptive defence-based multi-objective methodology that also adheres to NIST Cybersecurity Framework 2.0 guidelines. Moreover, based on the dependency of the system risk level, our proposed approach was a dynamic pipeline which can compromise between security and performance dynamically in runtime conditions and respond by varying attack treatment against GPU side-channel attacks (a Burstiness Complex Technology). Based on these remarks, we concluded that our contribution works could be considered significant GPU security contributions in protecting sensitive computations on GPU-accelerated environments for organisations.

VII. COMPARATIVE DISCUSSION

In this subsection, we compare our proposed secure shader execution framework with other existing models in terms of effectiveness, implementation complexity, and integration capability with the standard cyber security mechanism; for example, [7] focuses on timing attacks by leveraging constant time algorithms. However, they do not consider other side-channel attacks. In contributing to their work, our framework defends against timing attacks, power attacks, and cache attacks at once with the unified defence mechanism. Introduced power analysis attacks only. Our proposed framework defence against power analysis by adding a PA module into the timing and cache countermeasures-based protection approach.

Further, [38] We propose secure cache architectures to mitigate cache attacks, which may require significant hardware modifications. Our framework, utilising software-based secure

shader execution and runtime isolation techniques, provides a more practical solution with minimal hardware changes. It explores countermeasures like noise injection but at the cost of increased computational overhead. Our framework contributes by optimising shader execution patterns to balance security and performance effectively.

Other models, such as that of [39], present a holistic GPU security approach but lack a structured risk management strategy that aligns with industry standards. Our framework not only integrates technical countermeasures but also aligns with NIST's Cybersecurity Framework 2.0 for a comprehensive cybersecurity strategy. Among implementation complexity models, [40] provides simpler solutions targeting specific attack vectors, and [41] proposes more complex methodologies; however, our proposed model strikes the right balance between complexity and manageability based on recent studies' findings in this regard as well.

Runtime isolation via hardware-assisted approaches in our framework offers strong protection against compromised GPU drivers or kernels, which is usually not considered to this extent by other existing works; thus, we mean our framework is standard in terms of hardware security. To summarise, indeed, current research work provides GPU security contributions, but the adoption of a holistic, adaptive, and integrative approach by our framework to solve the problem is not comparable with any other work. Besides technical novelty, in the light of a predefined solid risk management plan, we maximise the level of security that can be achieved according to industry best practices for risk mitigation against GPU accelerated environment threats.

VIII. CONCLUSION AND FUTURE WORK

Our work is the only one that integrates the NIST Cybersecurity Framework 2.0 (NIST consensus document) as well as the secure shader execution framework to improve resilience against GPU side-channel attacks. Particularly, our work provides a holistic and dynamic solution by considering timing attack, power attack and cache attack simultaneously. The previous works are generally targeted for specific attacks, and their solutions cannot be converted into vulnerability-based frameworks easily.

We also use advanced mathematical modelling techniques coupled with real-time adaptability to react dynamically against possible unknown vulnerabilities and threats while ensuring that no major degradation in system performance occurs. Moreover, by aligning with the NIST framework, we encourage structured risk management and governance, which ensures an increase in overall organisational resilience when using our approach.

Future works will include expanding our framework on new side-channel attacks and adapting it with emerging technologies, e.g., quantum computing and the use of AI for better adaptability. Extensive field testing, guidelines development, and practice will be included to make this approach feasible. In addition, intra-regulatory policies and global regulatory compliance for technology adoption are needed.

To summarise, our model presents a solidified, adaptable and structured implementation of a defence mechanism against GPU side-channel attacks, which significantly enhances organisational cyber-security and resilience.

REFERENCES

- Scholl, M., & Suloway, T. (2022). Suloway, "Introduction to Cybersecurity for Commercial Satellite Operations," doi: 10.6028/nist.ir.8270-draft2.
- Tiemann, T., Weissman, Z., Eisenbarth, T., & Sunar, B. (2022). "IOTLB-SC: An Accelerator-Independent Leakage Source in Modern Cloud Systems," doi: 10.48550/arxiv.2202.11623.
- Lungu, N., Tembo, S., Patra, S. S., Walubita, N., Dash, B. B., & De, U. C. (2024, March). Probing Vulnerabilities in GPU Shader Execution. In *2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)* (pp. 1-6). IEEE. doi: 10.1109/dicct61038.2024.10532862.
- Lungu, N., Tembo, S., Sasmal, G. C., Rout, S., Gourisaria, M. K., & Patra, S. S. (2024, May). Precision Countermeasures for GPU Side Channels Using Shader Execution Pattern Analysis. In *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)* (pp. 1-6). IEEE. doi: 10.1109/aiiot58432.2024.10574735.
- Pascoe, C. E. (2023). Public Draft: The NIST Cybersecurity Framework 2.0.
- Hätinen, M. (2024). Evolution of cybersecurity risk management standardization: Comparison of the NIST Cybersecurity Framework version 1.1 and 2.0.
- Ge, Q., Yarom, Y., Cock, D., & Heiser, G. (2018). A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *Journal of Cryptographic Engineering*, 8, 1-27. doi: 10.1007/s13389-016-0141-6.
- Gamaarachchi, H., Ragel, R., & Jayasinghe, D. (2014, December). Accelerating correlation power analysis using graphics processing units (gpus). In *7th International Conference on Information and Automation for Sustainability* (pp. 1-6). IEEE. doi: 10.1109/ICIAFS.2014.7069547.
- D. Oswald, B. Paassen, and F. M'oller, (2020). "Side-channel analysis of wireless embedded systems: Cache attacks meet electromagnetic emanations," *ACM Trans Embed Comput Syst*, 19(2), 1-29.
- Liu, F., Yarom, Y., Ge, Q., Heiser, G., & Lee, R. B. (2015, May). Last-level cache side-channel attacks are practical. In *2015 IEEE symposium on security and privacy* (pp. 605-622). IEEE.
- Petersen, R., Santos, D., Smith, M., & Witte, G. (2020). *Workforce framework for cybersecurity (NICE*

- framework) (No. NIST Special Publication (SP) 800-181 Rev. 1 (Withdrawn)). National Institute of Standards and Technology. doi: 10.6028/nist.sp.800-181r1.
12. P. Wang, J. Liu, and Y. Zhang, (2020). "A survey of security threats and defences in GPU-accelerated systems," *J Parallel Distrib Comput*, vol. 136, pp. 203–217.
13. Paramasivam, S., PL, S. A., & Sathyamoorthi, P. (2022). Maximal overlap discrete wavelet transform-based power trace alignment algorithm against random delay countermeasure. *Etri Journal*, 44(3), 512-523. doi: 10.4218/etrij.2021-0074.
14. Lungu, N., Tembo, S., Walubita, N., & Patra, S. S. (2024, February). Mitigating GPU Side-Channels via Integrated Monitoring and Response. In *2024 International Conference on Integrated Circuits and Communication Systems (ICICACS)* (pp. 1-8). IEEE. doi: 10.1109/icicacs60521.2024.10498584.
15. Lungu, N. (2021). *Dos and memory exploitation sidebar attacks on graphics processing units on windows operating systems* (Doctoral dissertation, The University of Zambia).
16. W. Hua, H. Gao, T. Qin, M. T. Lazarescu, and S. Yao, (2020). "Side-channel analysis of cryptographic accelerators: A survey," *ACM Computing Surveys (CSUR)*, 53(5), pp. 1–30.
17. Zhang, C. (2024). State-of-the-Art Approaches to Enhancing Privacy Preservation of Machine Learning Datasets: A Survey. *arXiv preprint arXiv:2404.16847*. doi: 10.36227/techrxiv.170956580.07799070/v1.
18. Mell, P. (2011). The NIST Definition of Cloud Computing. *NIST Special Publication*, 800-145. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-145>
19. Kubota, T., Yoshida, K., Shiozaki, M., & Fujino, T. (2021). Deep learning side-channel attack against hardware implementations of AES. *Microprocessors and Microsystems*, 87, 103383. doi: 10.1016/j.micpro.2020.103383.
20. H. Naghibijouybari, K. N. Khasawneh, and N. Abu-Ghazaleh, (2019). "Mitigating Side-Channel Attacks on GPUs," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 166–177.
21. Bose, T. Kundu, and D. Mukhopadhyay, (2021). "Analysis of Side-Channel Attacks on GPUs: A Survey," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1470–1489.
22. Jeon, H., Karimian, N., & Lehman, T. (2021, April). A new foe in GPUs: Power side-channel attacks on neural network. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)* (pp. 313-313). IEEE. doi: 10.1109/ISQED51717.2021.9424358.
23. Park, J., Cho, H., Jung, W., & Lee, J. (2018). Transparent GPU memory management for DNNs. *ACM SIGPLAN Notices*, 53(1), 411-412. doi: 10.1145/3178487.3178531.
24. Mei Yin, D. B., Kamal, M. I., Azmanuddin, N. S., Ali, S. H. S., Othman, A. T., & Chik, R. Z. W. (2016, January). Electronic door access control using MyAccess two-factor authentication scheme featuring near-field communication and eigenface-based face recognition using principal component analysis. In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication* (pp. 1-8). doi: 10.1145/2857546.2857548.
25. Lungu, N., Banda, D., & Luka, N. (2023). SIDEBAR ATTACKS ON GPUS. *International Research Journal of Modernization in Engineering Technology and Science*, 5(2), 255-266. doi: 10.56726/IRJMETS33377.
26. Karimi, E., Fei, Y., & Kaeli, D. (2020, December). Hardware/software obfuscation against timing side-channel attack on a GPU. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* (pp. 122-131). IEEE. doi: 10.1109/HOST45689.2020.9300259.
27. Naghibijouybari, H., Neupane, A., Qian, Z., & Abu-Ghazaleh, N. (2018, October). Rendered insecure: Gpu side channel attacks are practical. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security* (pp. 2139-2153).
28. Luo, C., Fei, Y., Luo, P., Mukherjee, S., & Kaeli, D. (2015, October). Side-channel power analysis of a GPU AES implementation. In *2015 33rd IEEE International Conference on Computer Design (ICCD)* (pp. 281-288). IEEE. doi: 10.1109/ICCD.2015.7357115.
29. Khan, M. N. I., Bhasin, S., Liu, B., Yuan, A., Chattopadhyay, A., & Ghosh, S. (2021). Comprehensive study of side-channel attack on emerging non-volatile memories. *Journal of Low Power Electronics and Applications*, 11(4), 38. doi: 10.3390/jlpea11040038.
30. Chakrabarti, D., Rodgers, G., Bertolli, C., Bercea, G. T., Lehr, J. P., Stringer, L., ... & Lieberman, R. (2023, November). Specialized Kernels for Optimizing GPU Offload in OpenMP. In *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network,*

- Storage, and Analysis (pp. 1918-1928). doi: 10.1145/3624062.3624605.
31. Barthe, G., Grégoire, B., & Laporte, V. (2018, July). Secure compilation of side-channel countermeasures: the case of cryptographic “constant-time”. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)* (pp. 328-343). IEEE. doi: 10.1109/csf.2018.00031.
32. Bolchini, C., Cherubin, S., Durelli, G. C., Libutti, S., Miele, A., & Santambrogio, M. D. (2018). A runtime controller for OpenCL applications on heterogeneous system architectures. *ACM SIGBED Review*, 15(1), 29-35. doi: 10.1145/3199610.3199614.
33. Sakalis, C., Kaxiras, S., & Sjölander, M. (2021). Selectively Delaying Instructions to Prevent Microarchitectural Replay Attacks. *arXiv preprint arXiv:2103.10692*. doi: 10.48550/arxiv.2103.10692.
34. Power, J., Hestness, J., Orr, M. S., Hill, M. D., & Wood, D. A. (2014). gem5-gpu: A heterogeneous cpu-gpu simulator. *IEEE Computer Architecture Letters*, 14(1), 34-36. doi: 10.1109/LCA.2014.2299539.
35. Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J. W., Lee, S. H., & Skadron, K. (2009, October). Rodinia: A benchmark suite for heterogeneous computing. In *2009 IEEE international symposium on workload characterization (IISWC)* (pp. 44-54). Ieee.
36. Danalis, A., Marin, G., McCurdy, C., Meredith, J. S., Roth, P. C., Spafford, K., ... & Vetter, J. S. (2010, March). The scalable heterogeneous computing (SHOC) benchmark suite. In *Proceedings of the 3rd workshop on general-purpose computation on graphics processing units* (pp. 63-74). doi: 10.1145/1735688.1735702.
37. Li, H., Rajbahadur, G. K., & Bezemer, C. P. (2024). Studying the Impact of TensorFlow and PyTorch Bindings on Machine Learning Software Quality. *ACM Transactions on Software Engineering and Methodology*. doi: 10.1145/3678168.
38. Dessouky, G., Gruler, A., Mahmood, P., Sadeghi, A. R., & Stapf, E. (2021). Chunked-cache: On-demand and scalable cache isolation for security architectures. *arXiv preprint arXiv:2110.08139*. doi: 10.48550/arxiv.2110.08139.
39. Ozga, W., Quoc, D. L., & Fetzer, C. (2021). Perun: Secure multi-stakeholder machine learning framework with GPU support. *arXiv preprint arXiv:2103.16898*. doi: 10.48550/arxiv.2103.16898.
40. Mahmoud, D. G., Lenders, V., & Stojilović, M. (2022). Electrical-level attacks on CPUs, FPGAs, and GPUs: Survey and implications in the heterogeneous era. *ACM Computing Surveys (CSUR)*, 55(3), 1-40. doi: 10.1145/3498337.
41. Mittal, S., Abhinaya, S. B., Reddy, M., & Ali, I. (2018). A survey of techniques for improving security of gpus. *Journal of Hardware and Systems Security*, 2, 266-285. doi: 10.1007/s41635-018-0039-0.