Side-Channel Attack Detection using gem5 and Machine Learning: A Case Study on Fault-based Attacks in RISC-V

Mahreen Khan, Maria Mushtaq, Renaud Pacalet, Ludovic Apvrille

Side-Channel Attack Detection using gem5 and Machine Learning: A Case Study on Fault-based Attacks in RISC-V

Mahreen Khan
Telecom Paris
Institut Polytechnique de Paris
Palaiseau, France
mahreen.khan@telecom-paris.fr

Maria Mushtaq
Telecom Paris
Institut Polytechnique de Paris
Palaiseau, France
maria.mushtaq@telecom-paris.fr

Renaud Pacalet
Telecom Paris
Institut Polytechnique de Paris
Palaiseau, France
renaud.pacalet@telecom-paris.fr

Ludovic Apvrille
Telecom Paris
Institut Polytechnique de Paris
Palaiseau, France
ludovic.apvrille@telecom-paris.fr

***Abstract*—Microarchitectural side-channel attacks pose a significant threat to modern computing architectures. This paper presents a machine learning-based methodology for detecting these attacks using the gem5 simulator, focusing on the recently discovered Flush+Fault attack [6] on RISC-V. Our approach follows a three-phase process. The first phase is data collection, where we simulate attack and non-attack scenarios in gem5 and extract microarchitectural features indicative of side-channel activity. The second phase is the training phase, where we utilize machine learning (ML) techniques to build a classification model capable of distinguishing between normal execution and attack patterns. The last phase is the testing phase, where we evaluate the trained model using various performance metrics to validate its accuracy and precision. To the best of our knowledge, this is the first detection framework for Flush+Fault attacks [6] on RISC-V, showcasing its effectiveness in mitigating emerging threats. Our results indicate that gem5 metrics combined with machine learning models can reliably detect Flush+Fault attacks, achieving 0.99 accuracy with random forest (RF), 0.96 with support vector machine (SVM), and 0.95 with naïve bayes (NB). Moreover, this methodology is adaptable to different side-channel attacks and architectures, making it a promising approach for strengthening microarchitectural security.**

***Index Terms*—Side-channel attacks, fault-based attacks, machine learning, gem5, RISC-V, flush+fault attack, security, detection, vulnerability assessment, microarchitectural security, hardware security, anomaly detection.**

I. Introduction

Microarchitectural side-channel attacks (SCAs) exploit unintended hardware-level information leakage, posing significant threats to computational security. Modern processors, optimized for performance via speculative execution and caching, inadvertently introduce vulnerabilities exploited by attacks like Spectre, Meltdown, and various cache-based SCAs [10], [23]. While RISC-V is celebrated for its modular and open-source design, it is also susceptible to such threats. Notably, the Flush+Fault attack [6] targets RISC-V's instruction cache by combining cache flushing and fault-inducing behavior to infer sensitive data, underscoring the need for effective detection mechanisms tailored to RISC-V. Although prior work demonstrates the success of machine learning (ML) in detecting SCAs on x86 and ARM platforms [9], [13], ML-based detection using gem5 simulation for RISC-V remains underexplored.

This paper introduces a methodology for detecting microarchitectural SCAs across architectures (x86, ARM, RISC-V) by integrating gem5 simulation [12] with ML techniques. While our approach is architecture-agnostic, we validate it through a case study on the Flush+Fault attack in RISC-V, which is both novel and currently lacks detection solutions. Gem5 provides detailed simulation of microarchitectural events, such as cache activity and branch prediction, under both benign and attack scenarios. We extract critical performance metrics from these traces, then train an ML model to distinguish attacks from normal behavior.

Our key contributions are:

1) A generalized ML-based detection framework using gem5 to profile and detect various SCAs (cache-based, speculative execution) across x86, ARM, and RISC-V.
2) The first implementation and evaluation of an ML-based detection method for Flush+Fault attacks on RISC-V.

The rest of the paper is organized as follows: Section II reviews foundational concepts and related work. Section III describes our ML and gem5-based detection framework. Section IV evaluates its performance through a RISC-V Flush+Fault case study. Section V discusses broader implications and future directions. Section VI concludes the paper.

II. Preliminaries and Related Work

A. Flush+Fault Attack

The Flush+Fault attack [6] exploits timing variations due to instruction cache (I-cache) flushing and fault handling. Beyond cache eviction, it leverages exceptions to infer execution characteristics.

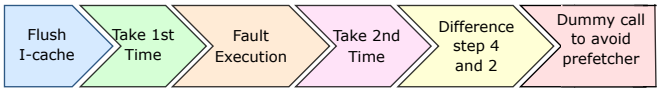

Fig. 1. Flush+Fault attack mechanism.

As illustrated in Fig. 1, the attacker begins by flushing the I-cache using RISC-V's `fence.i` to ensure future instruction fetches come from main memory. An initial timestamp is recorded using a high-resolution timer. Rather than executing victim code directly, the attacker jumps to an invalid address to trigger a fault via illegal or privileged instructions in user mode. This causes the CPU to fetch and decode the instruction, with timing depending on whether the line was cached.

The exception is intercepted via a signal handler, allowing a second timestamp to be recorded. The timing delta indicates cache status: cached instructions fault faster, resulting in lower latency. The attacker repeatedly jumps to a dummy address outside the target cache line to avoid speculative prefetching by the branch predictor. This ensures the target line isn't prefetched, preserving the critical timing leakage [6].

B. gem5 for Microarchitectural Attack Simulation

The gem5 simulator [12] offers a configurable platform for analyzing side-channel attacks. Supporting ISAs like x86, ARM, and RISC-V, it allows researchers to simulate attacks in syscall emulation (SE) mode or full-system (FS) mode. Its flexibility in choosing CPU models and memory subsystems makes it ideal for microarchitectural analysis.

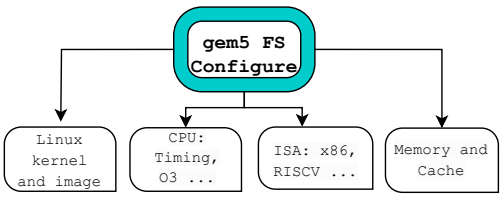

Fig. 2. Configuring gem5: ISA, CPU model, and memory hierarchy

Compiled attack binaries execute in the configured environment, where gem5 collects metrics like cache hit/miss rates, branch predictor stats, memory latency, and execution times—essential for evaluating attacks and discovering vulnerabilities.

While gem5 provides detailed behavior modeling, its simulation speed is slower than real hardware, and its timing precision may not exactly reflect real-world performance. Still, it remains a key tool in microarchitectural security research, with ongoing efforts to improve fidelity and efficiency.

C. Machine Learning Models

Machine learning enhances side-channel detection by classifying performance data from attacks. Random Forest (RF) [7] is an ensemble method that builds multiple decision trees from bootstrapped data and features. It excels in high-dimensional data environments and is robust to noise, making it effective for side-channel classification. Support Vector Machines (SVM) [4] construct optimal hyperplanes to separate classes and use kernel functions to handle non-linear data. Their generalization ability makes them suitable for detecting subtle differences in execution traces [11], [18]. Naïve Bayes (NB) [21] is a probabilistic classifier based on Bayes' theorem, assuming feature independence. Its simplicity and low computational cost make it attractive for environments with limited resources.

D. Related Work

Microarchitectural vulnerabilities—such as cache timing and speculative execution—have enabled powerful attacks like Meltdown [10] and Spectre [8], which exploit transient execution to leak sensitive data. Cache-based attacks, including Flush+Reload [23], further demonstrate the efficacy of side-channel exploitation. In the RISC-V ISA, the open-source nature has introduced unique threats such as the Flush+Fault attack [6], which combines cache flushing and fault-induced speculation to extract data.

Detection of microarchitectural side-channel attacks (SCAs) has been widely studied, often using hardware performance monitoring and machine learning (ML) for anomaly detection. Chiappetta et al. [3] were among the first to apply ML to detect Flush+Reload attacks. Payer's HexPADS [19] targeted Flush+Reload and Prime+Probe techniques. Zhang et al. [24] developed CloudRadar to monitor cache behavior in cloud environments. Briongos et al. introduced CacheShield [2], which actively tracks cache activity to detect various cache attacks. Mushtaq et al. proposed NIGHTs-WATCH [14] and WHISPER [16], classifying execution traces as benign or malicious.

Most of these works leverage hardware performance counters (HPCs), which monitor events such as cache misses, retired instructions, and branch mispredictions. These counters are specialized registers in the processor's Performance Monitoring Unit (PMU). Alam et al. [1] profiled branch prediction and cache attacks using HPCs, identifying key metrics like branch mispredictions and LLC misses as strong indicators of anomalous behavior.

However, HPC-based detection faces accuracy issues due to counter multiplexing, as discussed by Mushtaq et al. [15]. Although processors support many logical counters, only a few physical ones exist, requiring time-slicing that leads to outdated data, inaccuracies, and higher false positives. Furthermore, accessing HPCs adds system overhead, which can pollute caches and alter normal execution patterns.

To overcome these limitations, gem5 provides a more robust alternative by allowing precise, non-multiplexed tracking of microarchitectural events. Unlike hardware-bound HPCs,

gem5 offers fine-grained, architecture-independent simulation, improving the reliability of side-channel analysis.

This paper introduces a generic detection methodology integrating gem5 simulation with ML. In contrast to prior work relying on hardware counters, our approach extracts rich traces from gem5 for flexible and reproducible analysis. Furthermore, we present the first detection study of Flush+Fault attacks on RISC-V, filling a critical gap in microarchitectural security research.

III. Methodology for Side-Channel Attack Detection using gem5 and Machine Learning

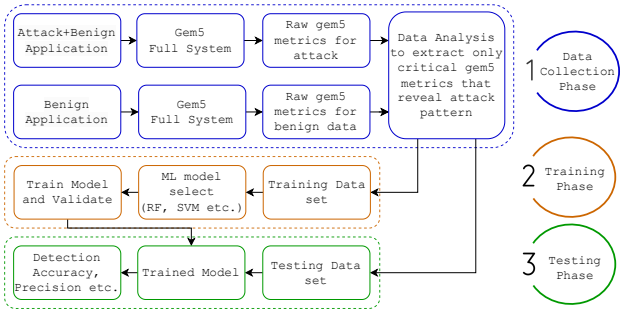

Fig. 3. Methodology for Side-Channel Attack Detection using gem5 and ML

Our approach for side-channel attack (SCA) detection combines microarchitectural tracing in gem5 [12] with machine learning (ML) classification. It comprises three phases: data collection, model training, and testing. This generalizable method supports architectures like x86, ARM, and RISC-V and targets attacks such as Flush+Reload and Spectre [8].

A. Data Collection Phase

We simulate both benign and attack scenarios in gem5, logging microarchitectural events such as cache misses, execution cycles, and branch mispredictions. These events reveal execution-dependent anomalies, including timing variations and speculative path behavior, which are indicative of SCAs [8], [23].

The resulting traces form a raw dataset that undergoes preprocessing and feature selection. We focus on discriminative features (e.g., cache miss rate, retired instructions, mispredictions) to capture subtle behavioral differences between normal and malicious execution.

B. Training Phase

Using the preprocessed dataset, we train classifiers (e.g., decision trees, SVMs, neural networks [20]) to distinguish between attack and benign patterns. Data is split into training and validation sets, with hyperparameters optimized via grid search or Bayesian methods [22]. The trained model learns to identify microarchitectural signatures of SCAs with high generalization.

C. Testing Phase

The final model is evaluated on a separate test set using standard metrics:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

High accuracy, precision, and recall ensure the classifier can reliably detect attacks while minimizing false alarms.

In summary, our methodology leverages gem5's fine-grained tracing with ML-based analysis to detect side-channel attacks across multiple platforms with robustness and scalability.

IV. Flush+Fault Attack Detection using gem5 and ML

In this paper, we address the detection of Flush+Fault [6] side-channel attacks in RISC-V by employing our proposed methodology. Our methodology leverages machine learning techniques to identify malicious activity using microarchitectural data simulated through gem5.

A. Data Collection Phase

The gem5 simulator was configured in full-system mode with the RISC-V 64-bit ISA and an out-of-order execution model using the `O3CPU` [12]. The system ran on the `riscv-bootloader-vmlinux-5.10` kernel [5], with modifications to the RISC-V disk image [5] to include the attack binary, ensuring a realistic simulation environment for side-channel analysis. Table I summarizes the experimental setup and parameters for detecting the Flush+Fault attack using our generic methodology.

TABLE I
Experimental Setup for Flush+Fault Attack Detection.

Parameter	Details
Attack	Flush+Fault
Simulator	gem5 full system
Metrics	I-cache miss, Branch mispredict
ML model	RF, SVM, NB
Data set	Training (0.8), Testing (0.2)

To detect Flush+Fault attacks, we collected a comprehensive set of microarchitectural metrics during gem5 simulations. These metrics were chosen for their sensitivity to deviations in processor behavior under attack conditions. The two key metrics were:

- **Total Branch mispredictions:** Flush+Fault attacks interfere with speculative execution, leading to an increased number of mispredictions, making this a critical indicator of attack behavior.

- **Total instruction cache misses:** Frequent cache flushes in attack scenarios cause a rise in instruction cache misses, which can serve as another reliable attack pattern.

Fig. 4 and Fig. 5 illustrate total branch mispredictions and total instruction cache misses in attack and non-attack scenarios, respectively.

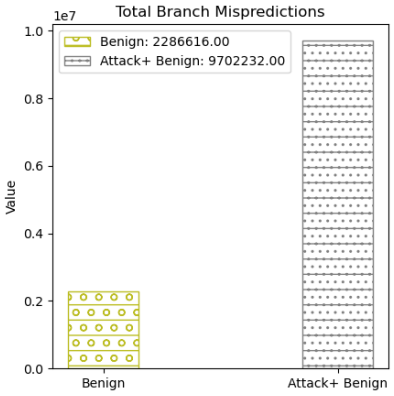

Fig. 4. Branch mispredictions in attack and non-attack scenarios.

Data was collected under moderate system load for both attack and benign executions. Once gathered, the dataset was preprocessed, with extracted performance metrics serving as input features and attack labels assigned in the final column. The dataset was then split into training (0.8) and testing (0.2) subsets to enable model training and evaluation.

B. Training and Testing Phase

For detection, we trained three machine learning models: random forest (RF) [7], support vector machine (SVM) [4], and naïve bayes (NB) [21]. By leveraging carefully selected gem5 performance metrics, our approach ensures accurate identification of attack scenarios. This optimization is essential for practical deployment, as it improves detection reliability and reduces unnecessary security interventions.

To evaluate generalization capabilities, we tested the trained models on an independent dataset containing unseen attack traces. As summarized in Table II, RF achieved the highest accuracy at 0.99, followed by SVM (0.96) and NB (0.95). RF demonstrated exceptional reliability, ensuring near-perfect detection with minimal error. SVM also performed well, making it a strong alternative.

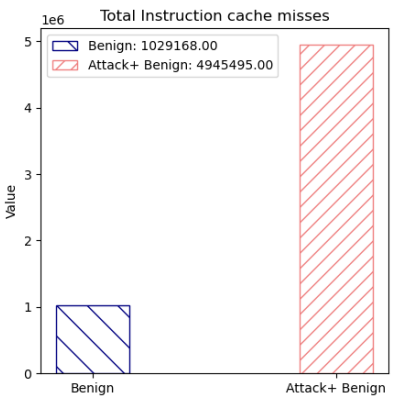

Fig. 5. Total Instruction Cache Misses in attack and non-attack scenarios.

Naïve Bayes, while slightly less accurate than RF and SVM, still achieved 0.95 accuracy, making it a viable, lightweight option. Unlike SVM and RF, which rely on more complex decision boundaries, NB's probabilistic approach allows for faster classification, making it suitable for resource-constrained detection scenarios.

These results emphasize the importance of selecting the right combination of gem5 metrics and machine learning models to build an effective side-channel attack detection framework.

TABLE II
FLUSH+FAULT DETECTION RESULTS USING RF, SVM, AND NB.

Model	Accuracy	Precision	Recall
RF	0.99	0.99	0.99
SVM	0.96	0.95	0.97
NB	0.95	0.92	0.96

V. DISCUSSION AND FUTURE WORK

Our results show that machine learning models can detect Flush+Fault attacks on RISC-V with high accuracy. Using gem5-based full-system simulation, we extracted microarchitectural features indicative of side-channel activity. The random forest model achieved 0.99 accuracy, demonstrating the effectiveness of combining gem5 with ML for microarchitectural threat detection.

Though focused on Flush+Fault, our methodology is adaptable to other attacks, including cache-based [17], [23] and speculative execution vulnerabilities [8]. Future work should extend this framework to diverse attack types and architectures like ARM and x86 to validate its generality.

A promising direction is transforming gem5 into a full-fledged security research platform with configurable cache and pipeline templates, integrated profiling, and performance visualization tools. Currently, raw data is processed offline through statistical analysis. Future research should explore real-time detection by integrating ML directly into the monitoring pipeline to enable proactive defense.

VI. CONCLUSION

We presented a machine learning-based framework for detecting side-channel attacks using the gem5 simulator [12]. By analyzing gem5-derived microarchitectural metrics under attack and non-attack conditions, we trained classifiers to distinguish malicious behavior. This work is the first to detect Flush+Fault attacks [6] on RISC-V, achieving high accuracy: 0.99 (RF), 0.96 (SVM), and 0.95 (NB). The proposed method is adaptable to various side-channel attacks and architectures, laying a strong foundation for advanced microarchitectural threat detection through simulation and machine learning.

REFERENCES

[1] M. Alam, S. Bhattacharya, D. Mukhopadhyay, and S. Bhattacharya. Performance counters to rescue: A machine learning based safeguard against micro-architectural side-channel-attacks. *Cryptology ePrint Archive*, 2017.

[2] S. Briongos, G. Irazoqui, P. Malagón, and T. Eisenbarth. CacheShield: Detecting cache attacks through self-observation. In *Proc. 8th ACM Conf. Data Appl. Secur. Privacy*, pages 224–235, Mar 2018.

[3] M. Chiappetta, E. Savas, and C. Yilmaz. Real time detection of cache-based side-channel attacks using hardware performance counters. *Appl. Soft Comput.*, 49:1162–1174, Dec 2016.

[4] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

[5] gem5 Contributors. The gem5 simulator. https://resources.gem5.org/, 2023.

[6] L. Gerlach, D. Weber, R. Zhang, and M. Schwarz. A security risc: microarchitectural attacks on hardware risc-v cpus. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2321–2338. IEEE, 2023.

[7] T. K. Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[8] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, et al. Spectre attacks: Exploiting speculative execution. *Communications of the ACM*, 63(7):93–101, 2020.

[9] L. Lerman, G. Bontempi, and O. Markowitch. A machine learning approach against a masked aes: Reaching the limit of side-channel attacks with a learning model. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 29–47. Springer, 2015.

[10] M. Lipp, M. Schwarz, D. Gruss, et al. Meltdown: Reading kernel memory from user space. In *USENIX Security Symposium*, pages 973–990. USENIX, 2018.

[11] Y. Liu and F. Koushanfar. Machine learning for side-channel analysis: A systematic review. *ACM Computing Surveys*, 54(6):1–34, 2021.

[12] J. Lowe-Power et al. The gem5 simulator: Version 20.0+. *arXiv preprint arXiv:2007.03152*, 2020.

[13] H. Maghrebi, T. Portigliatti, and E. Prouff. Deep learning-based side-channel analysis: A review. *ACM Computing Surveys*, 54(11s):1–36, 2021.

[14] M. Mushtaq, A. Akram, M. K. Bhatti, M. Chaudhry, V. Lapotre, and G. Gogniat. NIGHTs-WATCH: A cache-based side-channel intrusion detector using hardware performance counters. In *Proc. 7th Int. Workshop Hardw. Architectural Support Secur. Privacy*, page 1, Jun 2018.

[15] M. Mushtaq, P. Benoit, and U. Farooq. Challenges of using performance counters in security against side-channel leakage. In *CYBER 2020-5th International Conference on Cyber-Technologies and Cyber-Systems*, 2020.

[16] M. Mushtaq et al. WHISPER: A tool for run-time detection of side-channel attacks. *IEEE Access*, 8:83871–83900, 2020.

[17] M. Mushtaq, M. A. Mukhtar, V. Lapotre, M. K. Bhatti, and G. Gogniat. Winter is here! a decade of cache-based side-channel attacks, detection & mitigation for rsa. *Information Systems*, 92:101524, 2020.

[18] T. Nguyen and R. Karri. A comparative study of machine learning models for hardware trojan detection. *IEEE Transactions on CAD*, 39(12):4572–4585, 2020.

[19] M. Payer. HexPADS: A platform to detect 'stealth' attacks. In *Proc. Int. Symp. Eng. Secure Softw. Syst.*, pages 138–154, London, U.K., 2016. Springer.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] I. Rish. An empirical study of the naive bayes classifier. In *IJCAI-01 Workshop on Empirical Methods in Artificial Intelligence*, pages 41–46. Citeseer, 2001.

[22] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, 25:2951–2959, 2012.

[23] Y. Yarom and K. Falkner. Flush+reload: A high resolution, low noise, l3 cache side-channel attack. In *USENIX Security Symposium*, pages 719–732. USENIX, 2014.

[24] T. Zhang, Y. Zhang, and R. B. Lee. CloudRadar: A real-time side-channel attack detection system in clouds. In *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, pages 118–140, Paris, France, 2016. Springer.