Optimized Parallel Architectures of Post-Quantum Signature SPHINCS$^+$ on GPUs

Jiahao Xiang and Lang Li.

***Abstract*—The Post-Quantum Cryptography (PQC) standardization process has led to the development of SPHINCS$^+$, a stateless hash-based signature scheme that provides long-term security. The high computational cost of SPHINCS$^+$ has motivated research into efficient implementations on various platforms. In this work, we present a GPU-based implementation of SPHINCS$^+$ that achieves high throughput while maintaining security guarantees. Our implementation leverages the parallel processing capabilities of GPUs to accelerate the signature generation process. We evaluate the performance of our implementation on an NVIDIA RTX 4090 GPU and demonstrate that it can achieve a throughput of xxx for the SPHINCS$^+$ signature generation. Our results show that GPUs can be an effective platform for accelerating SPHINCS$^+$ and other post-quantum cryptographic schemes.**

***Index Terms*—Software implementation, GPU, signature algorithm.**

I. Introduction

THE quantum computers leverage quantum-mechanical phenomena to process data, raising significant concerns about the resilience of classical cryptographic methods. The security offered by widely deployed public-key cryptosystems, such as RSA and ECC, is jeopardized by Shor's algorithm [1], motivating comprehensive research on alternative cryptographic solutions. In response, the National Institute of Standards and Technology (NIST) initiated the Post-Quantum Cryptography (PQC) standardization process to develop novel schemes that withstand quantum computing capabilities [2].

SPHINCS$^+$ is a representative stateless hash-based signature scheme and a finalist in the ongoing NIST standardization effort [3]. Long-term security against advanced quantum attacks is targeted by employing robust cryptographic hash functions [4]. The high computational cost of SPHINCS$^+$ has motivated further investigations into efficient implementations across CPUs, FPGAs, and GPUs [5] to facilitate smooth adoption by organizations transitioning to post-quantum cryptography.

This work is supported by the Hunan Provincial Natural Science Foundation of China (2022JJ30103), Postgraduate Scientific Research Innovation Project of Hunan Province (CX20240977), "the 14th Five-Year Plan" Key Disciplines and Application-oriented Special Disciplines of Hunan Province (Xiangjiaotong [2022] 351), the Science and Technology Innovation Program of Hunan Province (2016TP1020).

Jiahao Xiang and Lang Li are affiliated with the Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, as well as the Hunan Engineering Research Center of Cyberspace Security Technology and Applications, both located at Hengyang Normal University, Hengyang 421002, China. They are also faculty members of the College of Computer Science and Technology at Hengyang Normal University. (e-mail: jiahaoxiang2000@gmail.com; lilang911@126.com)

A. Related Work

Recent years have witnessed significant progress in GPU-based implementations of SPHINCS$^+$. Lee and Hwang [6] pioneered the exploration of GPU acceleration for post-quantum cryptographic schemes, establishing foundational techniques for parallel implementation of hash-based signatures. Building upon this foundation, Kim et al. [7] introduced parallel methods for key components of SPHINCS$^+$, including FORS, WOTS$^+$, and MSS tree computations. Their implementation on an RTX 3090 GPU demonstrated significant throughput improvements, though it faced efficiency limitations due to multiple CUDA kernel launches.

Most recently, Wang et al. [8] presented CUSPX, introducing a comprehensive three-level parallelism framework that integrates algorithmic, data, and hybrid parallelization strategies. Their implementation incorporated novel parallel Merkle tree construction algorithms and multiple load-balancing approaches, achieving substantial performance improvements over previous implementations. Additionally, Ning et al. [9] proposed GRASP, which further optimized GPU-based SPHINCS$^+$ implementation through adaptive parallelization strategies and kernel fusion technology.

B. Motivation

While previous implementations have made significant strides in GPU acceleration of SPHINCS$^+$, several critical aspects remain unexplored. Existing approaches primarily focus on maximizing throughput through extensive parallelization, often overlooking the efficiency of individual thread execution. The implementation by Kim et al. [7] demonstrated the potential of parallel processing but was limited by multiple kernel launches. Although CUSPX [8] introduced a comprehensive parallelization framework, its approach to thread utilization could be further optimized.

Two key observations motivate our work. First, current implementations typically concentrate on parallelizing the SPHINCS$^+$ algorithm structure while paying less attention to the parallel optimization of underlying hash functions. A more holistic approach that considers both algorithmic levels could yield better performance. Second, existing implementations often prioritize maximum parallelism without fully considering the trade-off between thread count and execution efficiency. This can lead to suboptimal performance due to increased synchronization overhead and reduced work efficiency per thread.

These observations suggest the need for a more balanced approach that optimizes both the degree of parallelism and the

computational efficiency of individual threads. Our work aims to address these limitations by developing an implementation that not only leverages GPU parallelism effectively but also ensures efficient utilization of computational resources at the thread level.

C. Contributions

In this brief, an optimized GPU-based implementation of SPHINCS$^+$ is presented, achieving high throughput without compromising security. The main contributions are summarized as follows:

1) A novel parallelization strategy is introduced that balances the degree of parallelism with the computational efficiency of individual threads, thereby enhancing GPU resource utilization.
2) The parallel architectures of SPHINCS$^+$ are optimized by integrating algorithmic and data-level parallelization techniques, which improve the performance of the underlying hash functions.
3) The implementation is evaluated on an NVIDIA GPU, demonstrating a throughput of XXX SPHINCS$^+$ signatures per second, significantly exceeding the performance of state-of-the-art approaches. The complete source code and implementation details are available at https://github.com/jiahaoxiang2000/sphincs-plus.

The remainder of the brief is organized as follows. Section II provides an overview of the SPHINCS$^+$ signature scheme; Section III details the GPU-based implementation; Section IV presents the performance evaluation; and Section V concludes the brief.

II. Preliminaries

III. GPU-based Implementation of SPHINCS$^+$

IV. Performance Evaluation

V. Conclusion

References

[1] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

[2] National Institute of Standards and Technology, "Report on post-quantum cryptography," National Institute of Standards and Technology, Tech. Rep. NISTIR 8105, 2016. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf

[3] M. S. Turan, K. McKay, D. Chang, L. E. Bassham, J. Kang, N. D. Waller, J. M. Kelsey, and D. Hong, "Status report on the final round of the nist lightweight cryptography standardization process."

[4] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The sphincs$^+$ signature framework," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, L. Cavallaro, J. Kinder, X. Wang, and J. Katz, Eds. ACM, 2019, pp. 2129–2146.

[5] D. Joseph, R. Misoczki, M. Manzano, J. Tricot, F. D. Pinuaga, O. Lacombe, S. Leichenauer, J. Hidary, P. Venables, and R. Hansen, "Transitioning organizations to post-quantum cryptography," *Nat.*, vol. 605, no. 7909, pp. 237–243, 2022.

[6] W. Lee and S. O. Hwang, "High throughput implementation of post-quantum key encapsulation and decapsulation on GPU for internet of things applications," *IEEE Trans. Serv. Comput.*, vol. 15, no. 6, pp. 3275–3288, 2022.

[7] D. Kim, H. Choi, and S. C. Seo, "Parallel implementation of SPHINCS+ with gpus," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 71, no. 6, pp. 2810–2823, 2024.

[8] Z. Wang, X. Dong, H. Chen, Y. Kang, and Q. Wang, "Cuspx: Efficient gpu implementations of post-quantum signature sphincs¡sup¿+¡/sup¿," *IEEE Transactions on Computers*, vol. 74, no. 1, pp. 15–28, 2025.

[9] Y. Ning, J. Dong, J. Lin, F. Zheng, Y. Fu, Z. Dong, and F. Xiao, "GRASP: Accelerating hash-based PQC performance on GPU parallel architecture," Cryptology ePrint Archive, Paper 2024/1030, 2024. [Online]. Available: https://eprint.iacr.org/2024/1030