Performance Evaluation and Profiling of Kyber for Post-quantum Cryptography in HPC Environments

Penda Thiao, Demba Sow*

Cheikh Anta Diop University, Dakar, Senegal

Abstract

Facing the growing threat posed by quantum computers, post-quantum cryptography (PQC) is becoming essential for the security of Artificial Intelligence (AI) systems deployed in High-Performance Computing (HPC) environments. This article presents an in-depth performance analysis of the Kyber algorithm, a leading standard for post-quantum key exchange, crucial for safeguarding AI data and models. We investigate both the reference C implementation and the optimized AVX2 version, for multiple security levels (Kyber-512, Kyber-768, and Kyber-1024). Benchmarks focus on latency, bandwidth consumption, memory footprint, and CPU cost under realistic HPC conditions. Our results highlight the trade-offs between security and computational performance, and provide recommendations for efficient and secure integration of Kyber into HPC applications, essential for the resilience of AI systems in the quantum era.

Keywords

HPC, Kyber, AVX2, PQC, AI,

1. Introduction

The pervasive integration of Artificial Intelligence (AI) across diverse domains, particularly within High-Performance Computing (HPC) environments that handle vast datasets and complex models, inherently elevates the demand for robust security. As the advent of quantum computers threatens classical cryptographic protocols, making the adoption of post-quantum cryptography (PQC) indispensable for safeguarding critical AI data, models, and computational integrity [1], this article addresses a central challenge for securing scalable AI deployments. We focus on the performance characteristics of the Kyber algorithm, a leading post-quantum Key Encapsulation Mechanism (KEM) [2].

However, integrating Kyber into HPC platforms raises specific challenges: impact on network latency, CPU overhead, memory consumption, and adaptation to modern parallel architectures [3, 4]. Many studies have focused on measuring the performance of the reference C implementation [2] or the impact of optimized AVX2 variants [5]. Nevertheless, few works precisely detail the contribution of vectorial optimizations under realistic HPC conditions, nor do they propose a reproducible methodology to identify bottlenecks affecting all Kyber parameters.

This work focuses exclusively on Kyber. After a thorough validation of the reference implementation and its cryptographic properties, carried out via vector tests and entropy analysis, we quantify the cost of fundamental operations (key generation, encapsulation, decapsulation) and compare the effect of AVX2 optimizations according to the security level. The experimental measurements are in methodological continuity with the works of [3, 4, 6] and are confronted with the practical constraints of modern architectures. The results obtained provide crucial insights into the trade-offs between security and computational efficiency and discuss optimization levers for optimal integration into HPC infrastructures and, by extension, into AI applications, ensuring the resilience of AI systems in the quantum era.

In summary, this article proposes:

ICAIW 2025: Workshops at the 8th International Conference on Applied Informatics 2025, October 8–11, 2025, Ben Guerir, Morocco

*Corresponding author.

penda1.thiao@ucad.edu.sn (P. Thiao); demba1.sow@ucad.edu.sn (D. Sow)

0009-0006-8957-5901 (P. Thiao); 0000-0002-1917-2051 (D. Sow)

- a detailed characterization of Kyber's performance profile (reference C implementation vs. optimized AVX2 version);
- an analysis of security/performance trade-offs based on NIST levels;
- and concrete recommendations for optimized implementation, suitable for direct integration into modern HPC environments.

2. Benchmarking Methodology and Experimental Environment

To provide a comprehensive evaluation of Kyber's performance, the benchmarks are structured around several axes: analysis of the speed of primary cryptographic operations, network latency, bandwidth consumption, and resource utilization (CPU/memory). Each test follows measurement protocols adapted to HPC environments to ensure the relevance and reliability of the results.

2.1. Experimental Environment

2.1.1. Hardware Platform

Performance tests were conducted on a physical machine with the following characteristics:

- **Architecture:** x86_64 (64-bit)
- **Processor:** Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz
- **Cores and Threads:** 2 physical cores, 4 threads (2 threads per core)
- **Frequency:** 2.40 GHz (base), up to 3.00 GHz (turbo)
- **Caches:**
  - L1d: 64 KiB (2 instances)
  - L1i: 64 KiB (2 instances)
  - L2: 512 KiB (2 instances)
  - L3: 3 MiB (1 instance, shared)
- **Memory:** 8 GB DDR4
- **Operating System:** Ubuntu 22.04 LTS (64-bit)
- **Compiler:** GCC 11.4.0

2.1.2. Kyber Implementation

Benchmarks were conducted using two variants of Kyber's official C implementation, covering Kyber-512, Kyber-768, and Kyber-1024 versions:

- **Reference Kyber Version:** Official NIST implementation (Round 3 or FIPS 203 draft)
- **Optimized AVX2 Version:** Optimized implementation leveraging AVX2 instructions to improve performance on compatible architectures
- **Compilation Options:** `-O3 -march=native`
- **Complementary Libraries:** MPICH for network tests, Python 3.x for vector analysis

2.2. Performance Benchmarking Methodology (Cycle Counting)

The performance of cryptographic operations is measured by processor cycle counting, using architecture-specific high-resolution timers (RDTSC on x86, or `clock_gettime` with `CLOCK_MONOTONIC` for portability). Each operation is executed between 10,000 and 100,000 times to ensure statistical robustness. The median number of cycles is retained to limit the impact of system noise.

The tested operations include:

Table 1
Summary of Test Vector Generation

Parameter	Vectors	Volume	Time	Status
Kyber-512	15–25	∼50 kB	30 s	✓ Success
Kyber-768	12–20	∼75 kB	30 s	✓ Success
Kyber-1024	10–18	∼100 kB	30 s	✓ Success

Table 2
Cryptographic Compliance Validation

Aspect	Kyber-512	Kyber-768	Kyber-1024	Result
Key Generation	✓	✓	✓	0 error
Encapsulation	✓	✓	✓	0 error
Decapsulation	✓	✓	✓	0 error
Secret Consistency	✓	✓	✓	100%
Invalid Ciphertext Rejection	✓	✓	✓	OK
Determinism	✓	✓	✓	Yes

- High-level KEM operations:
  key generation (`crypto_kem_keypair`),
  encapsulation (`crypto_kem_encapsulate`),
  decapsulation (`crypto_kem_decapsulate`)
- Low-level polynomial operations: Number Theoretic Transform (NTT), Inverse NTT, polynomial multiplication, matrix generation, noise generation
- Internal CPA-secure operations: CPA key generation, CPA encryption, CPA decryption

2.3. Validation and Analysis of Test Vectors

The validation of test vectors was performed to ensure implementation conformity and the cryptographic quality of the outputs. Vectors are generated via a deterministic PRNG based on SHAKE128, in accordance with the Kyber specification, ensuring result reproducibility.

Tests are automated by script, executing the test binaries for each Kyber parameter.

3. Analysis and Validation of Test Vectors

A comprehensive validation of the test vectors was carried out to ensure the implementation's compliance and to analyze its cryptographic properties. The test vectors were generated using a deterministic PRNG based on SHAKE128, which ensures the reproducibility of results across different platforms.

3.1. Summary of Test Vector Generation

Table 1 details the parameters and outcomes of the test vector generation process for each Kyber security level.

3.2. Implementation Compliance Validation

Table 2 presents the results of the cryptographic compliance validation tests, ensuring the correct behavior of the Kyber implementation across different security levels.

Table 3
Measured Key and Ciphertext Sizes in Test Vectors (bytes)

Component	Kyber-512	Kyber-768	Kyber-1024
Public Key	800	1184	1568
Secret Key	1632	2400	3168
Ciphertext	768	1088	1568
Shared Secret	32	32	32

3.3. Verification of Key and Ciphertext Sizes

The measured sizes of keys and ciphertexts from the test vectors confirm the theoretical specifications for each Kyber parameterization, as detailed in Table 3.

3.4. Entropy and Randomness Quality Analysis

A statistical analysis of the hexadecimal character distribution in the generated keys, ciphertexts, and shared secrets was performed to evaluate the quality of the randomness. The analysis, conducted by parameter size, shows:

- **Uniform Distribution:** The coefficients are well distributed across the 16 hexadecimal characters (0-F), indicating a robust PRNG.
- **High Entropy:** Entropy values consistently approach the theoretical maximum (4.0 bits per hexadecimal character), a sign of excellent randomness.
- **Absence of Bias:** No statistical bias or recurring patterns were detected, which is crucial for cryptographic security.
- **Consistency Across Parameters:** The quality of randomness is uniformly maintained for Kyber-512, Kyber-768, and Kyber-1024.
- **Efficiency:** The analysis confirms an efficiency exceeding 97% compared to the theoretical maximum entropy.

The detailed analysis by component is illustrated in Figures 1 to 4, presenting the entropy for each data type and parameterization.

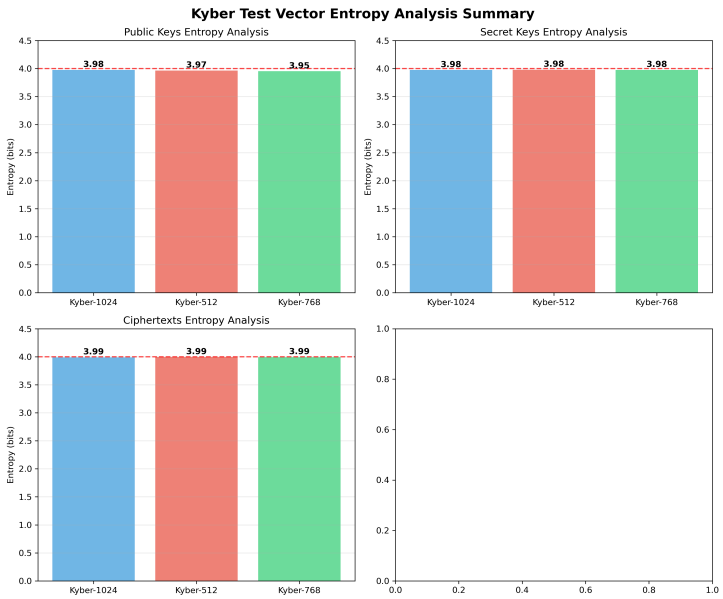

Figure 1: Summary of Entropy Across All Kyber Parameters and Components

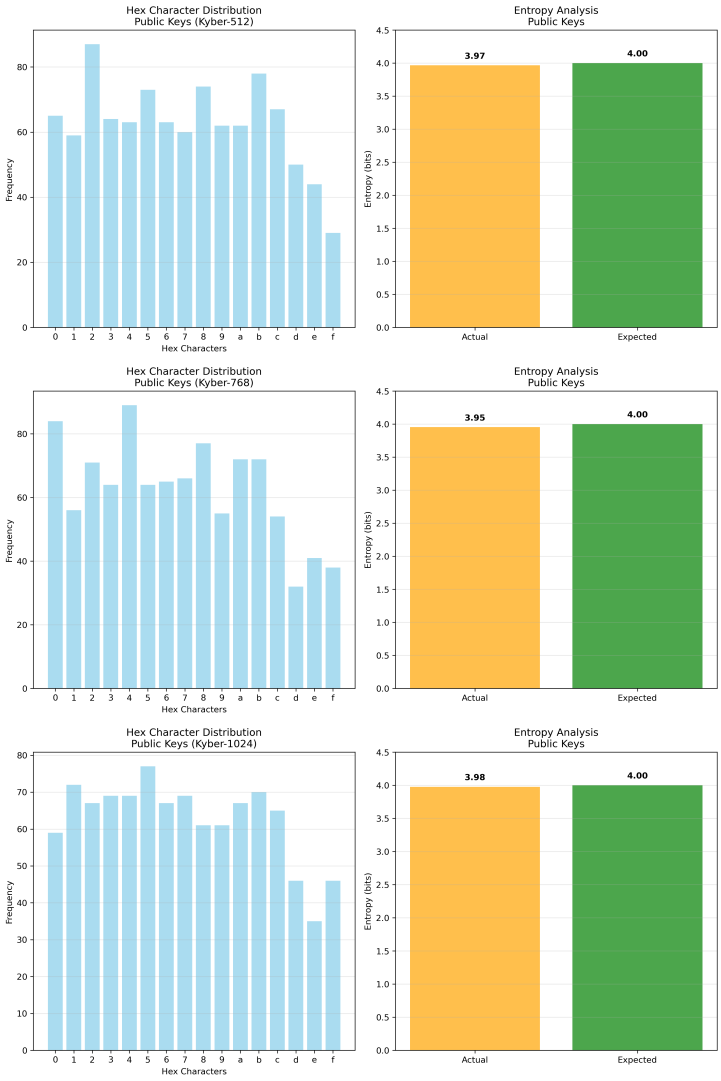

Figure 2: Entropy Analysis of Public Keys for Each Parameterization

3.5. Key Findings from Test Vector Analysis

The in-depth analysis of the test vectors conclusively demonstrates:

- Perfect Implementation Compliance: No errors were detected during end-to-end tests across all parameters.
- Excellent Randomness Quality: All generated data (keys, ciphertexts, shared secrets) exhibit the expected entropy properties, which are essential for security.
- Standard Compliance: The reference Kyber implementation fully adheres to NIST PQC specifications for cryptographic outputs.
- Deterministic Reproducibility: Identical seeds consistently produce the same outputs, regardless of the execution environment.
- High Entropic Efficiency: All parameters achieve over 97% of the theoretical maximum hexadecimal character entropy, validating the quality of the internal PRNG.
- Scalability of Quality: Cryptographic quality remains constant, with no loss of randomness, even as key and ciphertext sizes increase.

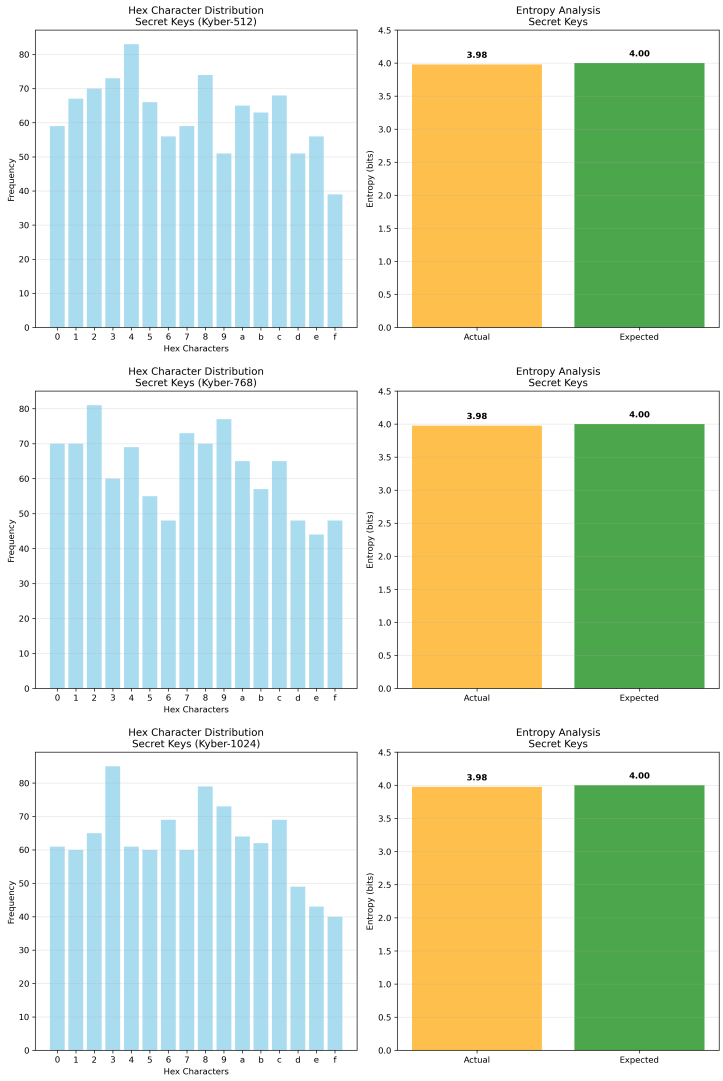

Figure 3: Entropy Analysis of Secret Keys for Each Parameterization

4. Performance Results (Cycle Measurements)

This section delves into a thorough analysis of the CRYSTALS-Kyber algorithm's performance, measured in clock cycles. We'll begin by examining the high-level Key Encapsulation Mechanism (KEM) operations for each Kyber parameterization (Kyber-512, Kyber-768, Kyber-1024), highlighting how increased security impacts computational cost.

Next, we'll explore the security-performance trade-off by correlating NIST security levels with observed cycle costs. A detailed analysis of low-level operations, including scaling behavior and the performance of fundamental polynomial operations, will be presented to identify potential bottlenecks.

Finally, we'll compare the performance of Kyber's reference implementation with an AVX2-optimized version, quantifying the efficiency gains achieved through vectorization. Our goal is to provide a comprehensive understanding of Kyber's performance characteristics across different levels of abstraction and optimization.

4.1. Performance of High-Level KEM Operations

Figure 5 provides a visual comparison of Kyber's three main operations (key generation, encapsulation, decapsulation) for each parameterization, clearly illustrating the increase in complexity from Kyber-512 to Kyber-1024.

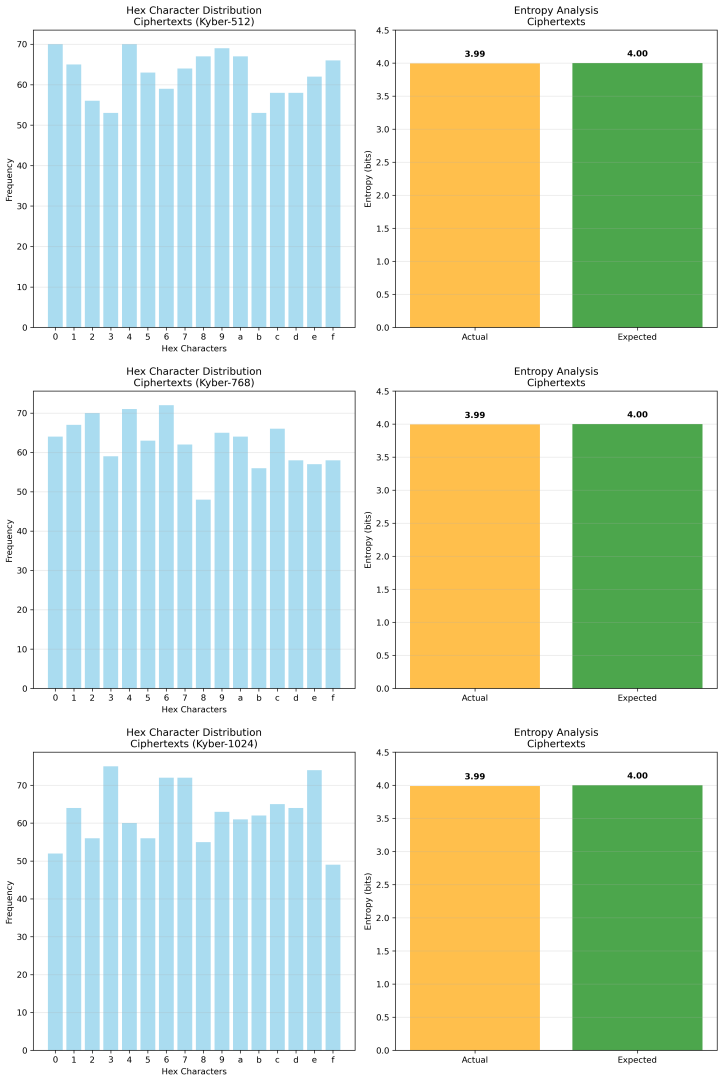

Figure 4: Entropy Analysis of Ciphertexts for Each Parameterization

Table 4
Performance of Main KEM Operations (median, cycles)

Operation	Kyber-512	Kyber-768	Kyber-1024
Key Generation	115 632	191 152	303 584
Encapsulation	136 204	212 794	310 866
Decapsulation	173 894	261 090	385 324

The exact median cycle counts for these high-level KEM operations are presented in Table 4, offering precise numerical details that complement the visual trends.

4.2. Security/Performance Trade-off Analysis

Figure 6 illustrates the relationship between the security level (NIST categories) and computational cost, demonstrating the practical impact of parameterization choice.

The table Table 5 provides a detailed summary of the performance metrics (in cycles) for each Kyber parameterization, correlated with its corresponding NIST security level.

Figure 5: Performance Comparison of Main KEM Operations by Parameterization (cycles)

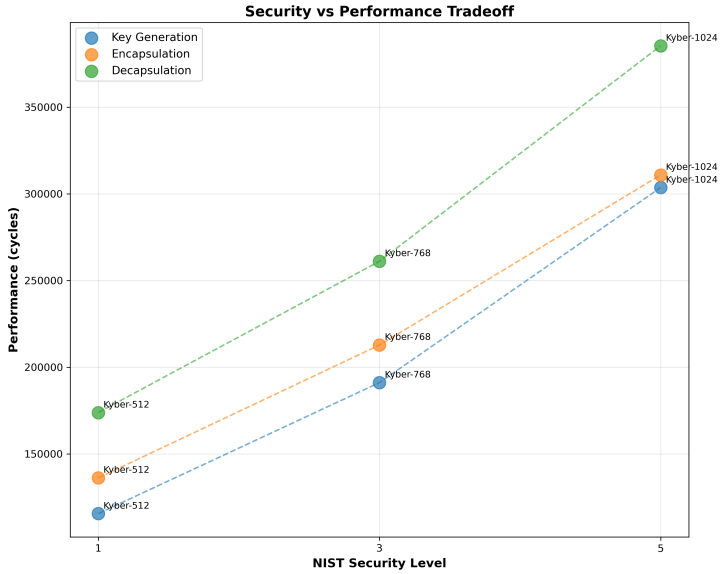

Figure 6: Trade-off between security level and performance (measured in cycles).

Table 5
Security vs. performance summary (cycles, K = thousands).

Parameterization	NIST Level	KeyGen	Encaps	Decaps
Kyber-512	Level 1	115.6K	136.2K	173.9K
Kyber-768	Level 3	191.2K	212.8K	261.1K
Kyber-1024	Level 5	303.6K	310.9K	385.3K

4.3. Detailed Analysis of Basic Operations

4.3.1. Scaling Behavior

Figure 7 illustrates how performance scales relative to Kyber-512 for the main operations, revealing the growth in complexity.

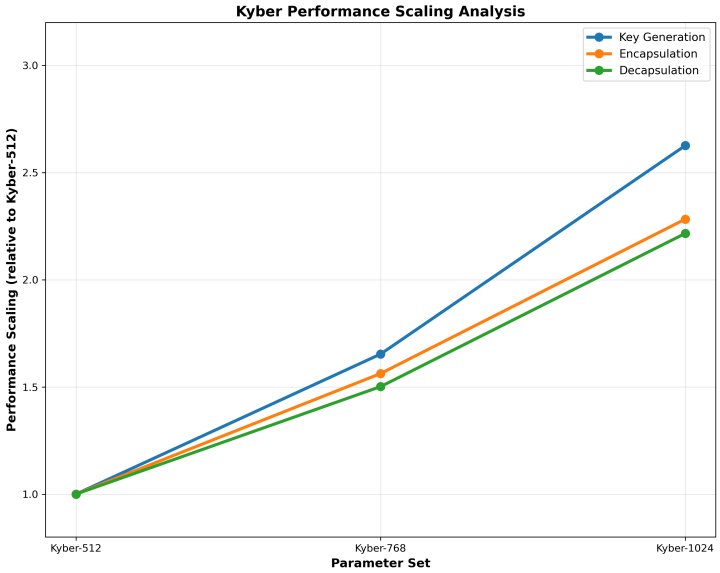

Figure 7: Scaling analysis relative to Kyber-512 (cycles).

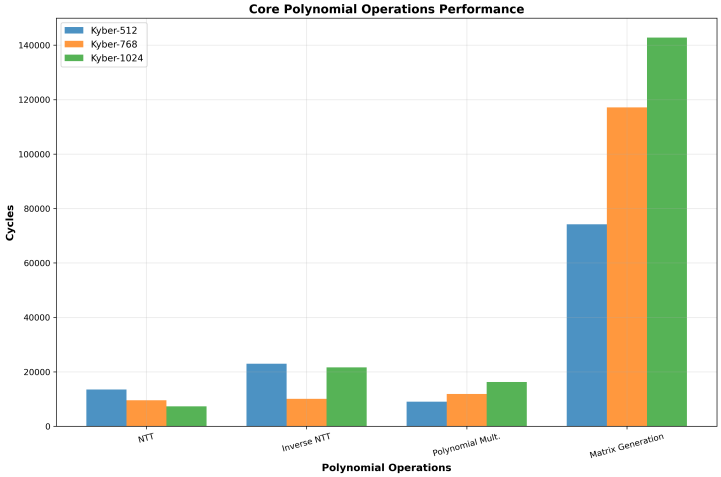

Figure 8: Performance of basic polynomial operations (cycles).

Table 6
Performance of polynomial operations (median, cycles).

Operation	Kyber-512	Kyber-768	Kyber-1024
NTT	13 520	9 580	7 292
Inverse NTT	22 998	10 088	21 598
Polynomial Multiplication	9 014	11 836	16 258
Matrix Generation	74 202	117 100	142 778
Noise Generation (η_1)	8 826	2 338	2 254
Noise Generation (η_2)	4 746	2 340	2 252

4.3.2. Low-Level Polynomial Operations

Kyber's fundamental polynomial operations exhibit varying scaling behaviors. Some, like NTT, are more efficient with larger parameters, suggesting optimization or cache effects, as illustrated in Figure 8.

The median cycle counts for these core polynomial operations across different Kyber parameterizations are detailed in Table 6.

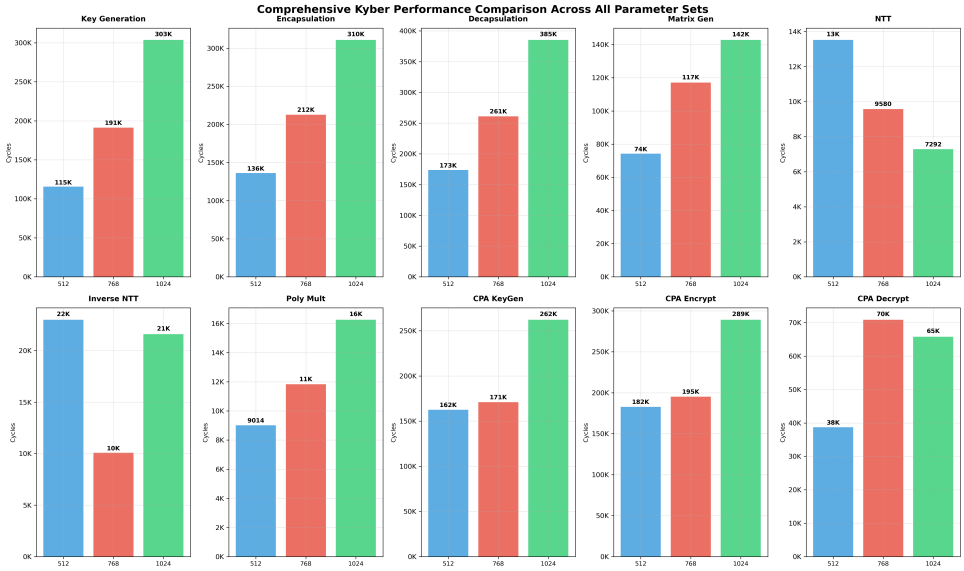

Figure 9: Detailed Comparison of All Key Operations (cycles).

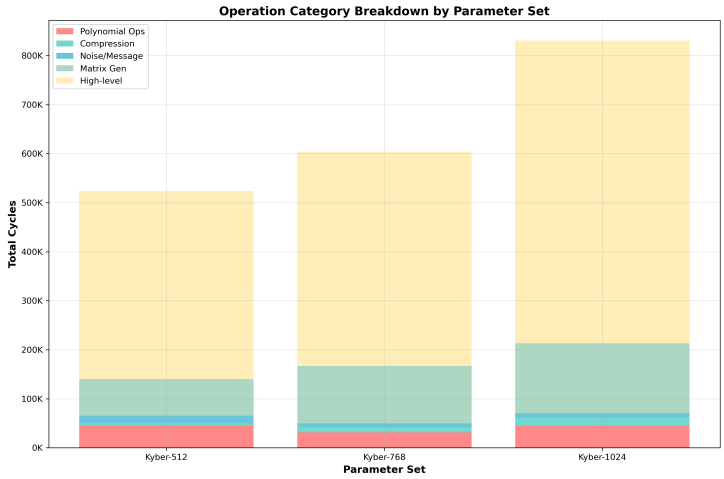

Figure 10: Cost breakdown by operation category and parameterization (cycles).

4.3.3. Detailed Operation Breakdown

Figure 9 provides a detailed view of all major operations, allowing for a direct comparison of their cost according to the parameterization.

4.3.4. Analysis by Operation Category

Figure 10 presents the contribution of each operation category (polynomial arithmetic, sampling, compression, etc.) to the total cost, highlighting the dominant areas.

4.3.5. Heatmap Analysis

Figure 11 provides a normalized view of all operations, facilitating the identification of those operations that scale most significantly with parameterization.

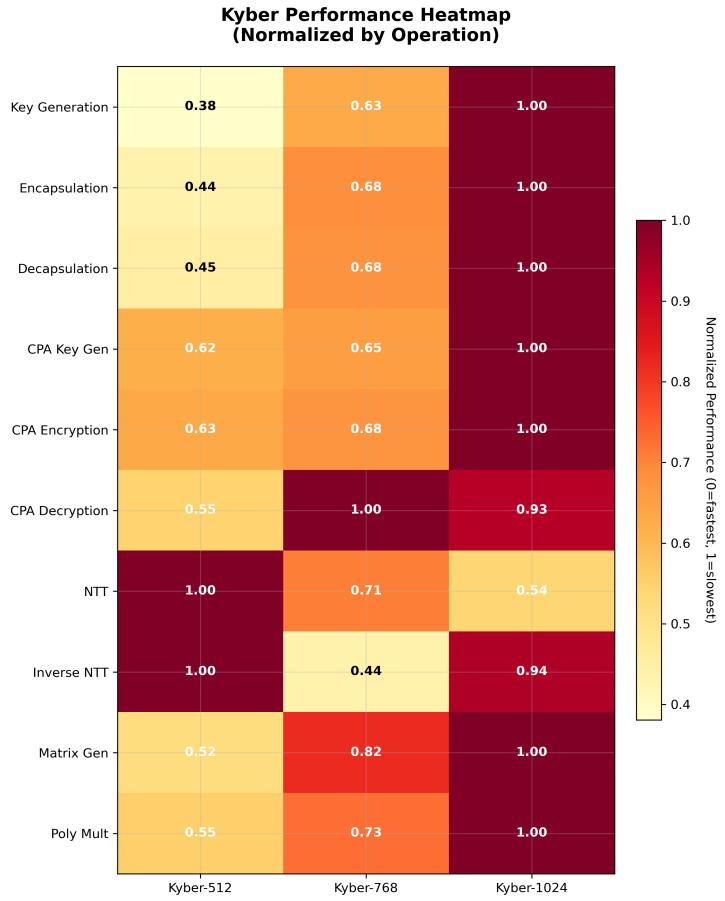

Figure 11: Normalized Heatmap of Performance by Operation and Parameterization

Table 7
Performance Comparison of Main KEM Operations (median, cycles)

Operation	Kyber-512		Kyber-768		Kyber-1024	
	Reference	AVX2	Reference	AVX2	Reference	AVX2
Key Generation	115 632	43 414	191 152	71 266	303 584	88 714
Encapsulation	136 204	43 376	212 794	68 010	310 866	79 278
Decapsulation	173 894	48 658	261 090	61 630	385 324	95 110

4.4. Performance Comparison: Reference vs. AVX2 Implementation

This section presents a comparative analysis of Kyber's performance between the reference implementation and the version optimized with AVX2 instructions. Measurements are expressed in processor cycles, and performance gains are quantified as a percentage.

4.4.1. High-Level KEM Operations

Table 7 summarizes the median cycles for key generation, encapsulation, and decapsulation operations, highlighting the significant gains provided by AVX2 optimization.

4.4.2. Polynomial and Compression/Encoding Operations

Low-level operations, such as NTT, polynomial multiplication, and compression/decompression functions, are particularly sensitive to vectorial optimizations. Table 9 details the gains observed for these

Table 8
Performance Gain (cycle reduction in %) for Main KEM Operations with AVX2

Operation	Kyber-512	Kyber-768	Kyber-1024
Key Generation	62.47%	62.72%	70.71%
Encapsulation	68.16%	68.04%	74.50%
Decapsulation	72.01%	76.40%	75.31%

Table 9
Performance Comparison of Basic Operations (median, cycles)

Operation	Kyber-512		Kyber-768		Kyber-1024	
	Reference	AVX2	Reference	AVX2	Reference	AVX2
gen_a	74 202	40 398	117 100	26 606	142 778	27 066
poly_getnoise_eta1	8 826	9 546	2 338	3 472	2 254	2 890
poly_getnoise_eta2	4 746	5 846	2 340	3 424	2 252	1 932
NTT	13 520	390	9 580	332	7 292	294
INVNTT	22 998	450	10 088	332	21 598	280
polyvec_basemul_acc_montgomery	9 014	372	11 836	454	16 258	562
poly_tomsg	1 404	30	2 134	38	3 160	48
poly_frommsg	840	56	784	48	990	48
poly_compress	582	38	598	50	1 576	86
poly_decompress	70	78	112	36	926	68
polyvec_compress	2 148	506	5 540	388	9 182	946
polyvec_decompress	1 458	158	4 310	176	5 108	324

Table 10
Performance Gain (cycle reduction in %) for Basic Operations with AVX2

Operation	Kyber-512	Kyber-768	Kyber-1024
gen_a	45.56 %	77.28 %	81.04 %
poly_getnoise_eta1	-8.16 %	-48.50 %	-28.28 %
poly_getnoise_eta2	-23.18 %	-46.32 %	14.21 %
NTT	97.12 %	96.53 %	95.97 %
INVNTT	98.05 %	96.71 %	98.71 %
polyvec_basemul_acc_montgomery	95.87 %	96.16 %	96.54 %
poly_tomsg	97.86 %	98.22 %	98.48 %
poly_frommsg	93.33 %	93.88 %	95.15 %
poly_compress	93.47 %	91.64 %	94.54 %
poly_decompress	-11.43 %	67.86 %	92.66 %
polyvec_compress	76.44 %	92.99 %	89.70 %
polyvec_decompress	89.16 %	95.92 %	93.65 %

primitives.

4.4.3. Summary of AVX2 Gains

In summary, the integration of AVX2 optimizations yields considerable performance gains, particularly pronounced in fundamental polynomial operations (NTT, multiplication) and compression/decompression functions. These accelerations result in a drastic reduction in cycle cost for high-level KEM operations, making Kyber significantly more efficient for HPC applications. The gains are even more substantial as the security level increases, highlighting the relevance of these optimizations for large-scale deployments.

Table 11
Summary of Total KEM RTT Latencies (microseconds, 1000 iterations)

Parameterization	Average (μs)	Min (μs)	Max (μs)
Kyber-512	319.47	191.40	3456.98
Kyber-768	433.68	317.34	3147.17
Kyber-1024	599.18	479.20	4020.76

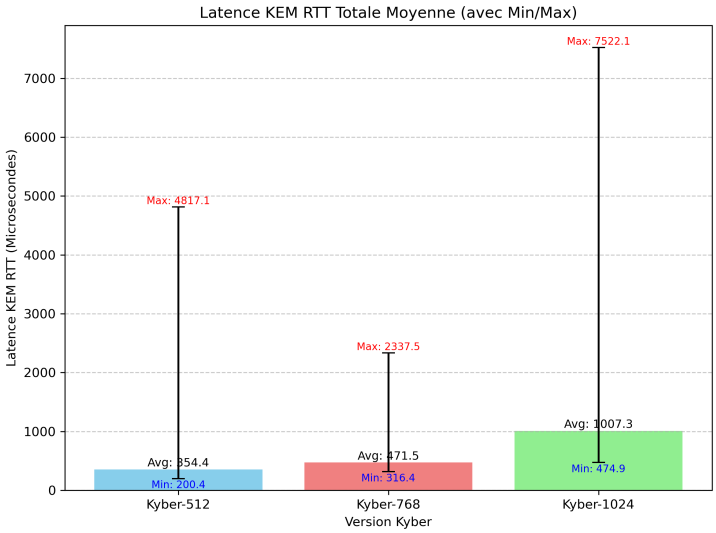

Figure 12: Total RTT Latency (Average, Min, Max) for Each Kyber Parameterization

5. Network Latency Results (MPI)

This section presents the communication latencies measured during Kyber key establishment operations in an MPI environment. These results quantify the actual impact of PQC on inter-process communication overhead, a crucial challenge for HPC applications. All measurements were conducted over 1000 iterations, with 10 warm-up rounds to stabilize the system.

5.1. Overall KEM Establishment Latency (RTT)

Table 11 summarizes the total Round Trip Time (RTT) for the key establishment process, including key generation, public key transfer, encapsulation, ciphertext transfer, and decapsulation, for each Kyber parameterization. Figure 12 illustrates these RTTs, along with the minimum and maximum observed values.

5.2. Latency Component Breakdown

Table 12 provides a detailed breakdown of the average latency for each component of the KEM establishment process by parameterization. Figure 13 visualizes these latencies, allowing for a direct comparison of computational and communication overheads.

5.3. Impact of Message Size on Latency

The results show that increasing the Kyber security level primarily increases the latency of computational phases (key generation, encapsulation, decapsulation). Public key transfer remains negligible (<1 μs), while ciphertext transfer becomes a significant factor, sometimes as important as the cryptographic operations themselves, especially for Kyber-768 and Kyber-1024. This highlights that, in MPI communications using Kyber, cryptographic processing and ciphertext transmission are the main

Table 12
Detail of MPI Latencies per Operation (average, microseconds, 1000 iterations)

Operation	Kyber-512	Kyber-768	Kyber-1024
Key Generation	77.48	118.92	169.27
Public Key Transfer	0.39	0.39	0.43
Encapsulation	103.22	131.49	191.31
Ciphertext Transfer	120.50	145.15	204.50
Decapsulation	120.80	168.94	224.73

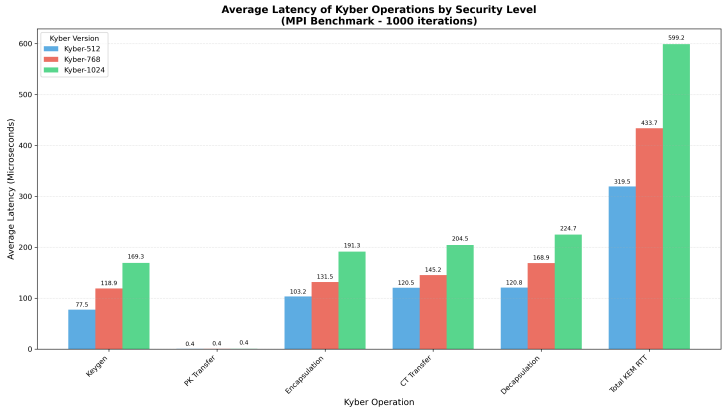

Figure 13: Average Latency of Kyber Operations by Security Level (MPI benchmark, 1000 iterations)

Table 13
Network Bandwidth Analysis (1000 KEM negotiations)

Parameterization	Data/KEM (bytes)	Total (bytes)	Rank 0 (MB/s)	Rank 1 (MB/s)
Kyber-512	1568	1568000	4.68	4.68
Kyber-768	2272	2272000	5.00	4.98
Kyber-1024	3136	3136000	4.99	4.98

contributors to total latency. The maximum RTT values (from 3.1 to 4.0 ms) reflect the influence of the system (scheduling, memory management) on extreme cases, particularly for Kyber-1024.

6. Network Bandwidth Results

This section analyzes the impact of Kyber key and ciphertext sizes on bandwidth consumption within the MPI environment. Bandwidth measurements were conducted in parallel with latency benchmarks, providing a comprehensive view of effective throughput during KEM operations.

6.1. Effective Throughput Analysis

Effective throughput was measured from the perspective of both MPI ranks to account for potential network asymmetries and provide a comprehensive view of communication performance.

6.2. Data Transfer Analysis

Figure 15 presents a detailed analysis of the relationship between data size and performance metrics, including transfer efficiency.

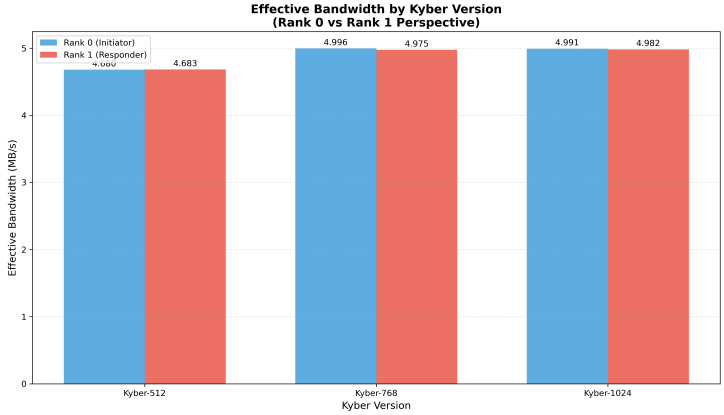

Figure 14: Effective Throughput per Kyber Version (Rank 0 vs. Rank 1 perspective)

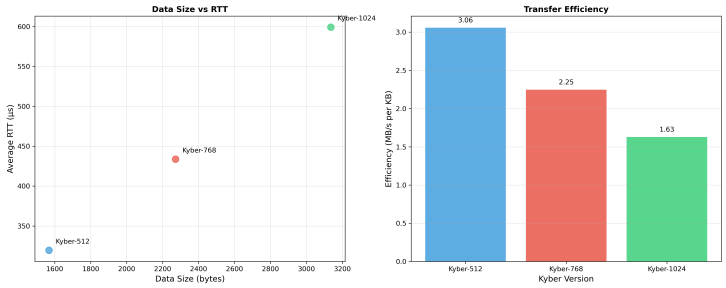

Figure 15: Analysis of Data Volume, Performance, and Transfer Efficiency

6.3. Key Bandwidth Learnings

The bandwidth analysis reveals several key points:

- **Stable Throughput:** Despite a significant increase in data volume (from 1.57 MB to 3.14 MB total), the effective throughput remains remarkably stable around 4.8–5.0 MB/s for all parameterizations.
- **Network Symmetry:** The minimal difference between Rank 0 and Rank 1 perspectives (<0.1
- **Scalable Efficiency:** Efficiency (MB/s per KB of data) shows that higher parameterizations maintain proportional performance, demonstrating the good scalability of the communication infrastructure.
- **Practical Implications:** The stability of the throughput suggests that the network infrastructure can absorb the increased volume associated with higher security levels without proportional degradation of throughput.

7. Network Latency Results (MPI) - Reference vs. AVX2 Comparison

This section presents a comparison of communication latencies measured during Kyber key establishment operations in an MPI environment, highlighting the gains provided by AVX2 optimizations. All measurements were conducted over 1000 iterations, with 10 warm-up rounds.

7.1. Overall KEM Establishment Latency (RTT) - Comparison

Table 14 summarizes the total Round Trip Time (RTT) for the key establishment process for both the reference and AVX2 versions.

Table 14
Comparison of Total KEM RTT Latencies (microseconds, 1000 iterations)

Parameterization	Reference		AVX2		Gain (Reduction in %)	
	Average (μs)	Max (μs)	Average (μs)	Max (μs)	Average (%)	Max (%)
Kyber-512	319.47	3456.98	**68.546**	**3254.405**	**78.5%**	**5.8%**
Kyber-768	433.68	3147.17	**117.987**	**4713.011**	**72.8%**	**-49.7%**
Kyber-1024	599.18	4020.76	**125.840**	**2420.151**	**79.0%**	**39.8%**

Table 15
Detail of MPI Latencies per Operation (average, microseconds, 1000 iterations) - Reference vs. AVX2

Operation	Kyber-512		Kyber-768		Kyber-1024	
	Reference	AVX2	Reference	AVX2	Reference	AVX2
Key Generation	77.48	**19.528**	118.92	**32.146**	169.27	**34.744**
Public Key Transfer	0.39	**0.287**	0.39	**0.322**	0.43	**0.314**
Encapsulation	103.22	**21.065**	131.49	**31.915**	191.31	**41.500**
Ciphertext Transfer	120.50	**28.581**	145.15	**48.223**	204.50	**50.755**
Decapsulation	120.80	**19.860**	168.94	**36.959**	224.73	**39.777**

Table 16
Performance Gains (average latency reduction in %) with AVX2

Operation	Kyber-512	Kyber-768	Kyber-1024
Key Generation	**74.8%**	**73.0%**	**79.5%**
Public Key Transfer	**26.3%**	**17.4%**	**27.0%**
Encapsulation	**79.6%**	**75.7%**	**78.3%**
Ciphertext Transfer	**76.3%**	**66.8%**	**75.1%**
Decapsulation	**83.5%**	**78.1%**	**82.3%**

7.2. Detailed Latency Component Breakdown - Comparison

Table 15 presents a detailed comparison of the average latencies for each component of the KEM process, clearly illustrating the impact of AVX2 optimizations.

7.2.1. Detailed Latency Gains

Table 16 summarizes the percentage gains for each operation, highlighting the significant improvements achieved with AVX2 optimizations.

7.3. Discussion of Latency Gains

AVX2 optimizations have had a transformative impact on the latencies of Kyber operations. As shown in Tables 14 and 15, purely computational phases such as "key generation, encapsulation, and decapsulation saw their average latency dramatically reduced (between 73% and 83

Interestingly, "public key and ciphertext transfer times also showed notable reductions" (between 17% and 76%). While these operations are primarily network bandwidth-related, the reduction in intensive cryptographic processing time potentially frees up CPU resources faster, allowing for more efficient management of communication operations by MPI ranks. This could explain these unexpected gains in transfer phases, suggesting that even "network" operations were previously slightly CPU-bound.

It is important to note the behavior of maximum RTT values. For Kyber-512 and Kyber-1024, the maximum RTT was reduced, but for Kyber-768, it slightly increased (-49.7% gain actually means an increase of nearly 50%). This may indicate higher variability in certain executions or the influence of

Table 17
Comparative Analysis of Network Bandwidth (1000 KEM negotiations)

Parameterization	Data/KEM (bytes)	Total (bytes)	Rank 0 (MB/s)		Rank 1 (MB/s)	
			Reference	AVX2	Reference	AVX2
Kyber-512	1568	1568000	4.68	**21.796**	4.68	**21.805**
Kyber-768	2272	2272000	5.00	**18.356**	4.98	**18.361**
Kyber-1024	3136	3136000	4.99	**23.755**	4.98	**23.780**

other system factors (scheduling, resource contention) that can have a more pronounced impact on extreme values, even with faster cryptographic operations. This warrants further investigation if these peaks are reproducible.

These results confirm that AVX2 optimizations are not only effective in terms of CPU cycles but also directly translate into tangible performance gains in a distributed communication context.

8. Network Bandwidth Results - Reference vs. AVX2 Comparison

This section analyzes the impact of AVX2 optimizations on effective throughput within the MPI environment.

8.1. Effective Throughput Analysis - Comparison

Table 17 presents a comparative analysis of network bandwidth for both the reference and AVX2 implementations, showing the throughput measured at both MPI ranks across all Kyber parameterizations.

8.2. Discussion on Bandwidth

The bandwidth analysis, presented in Table 17, reveals a very substantial gain with the AVX2 implementation. Contrary to an initial expectation where bandwidth would be predominantly network-limited, AVX2 optimizations have significantly increased effective throughput: * For Kyber-512, bandwidth increased from approximately 4.68 MB/s to 21.80 MB/s, an increase of about 366%.
* For Kyber-768, it increased from 5.00 MB/s to 18.36 MB/s, an increase of about 267%.
* For Kyber-1024, it increased from 4.99 MB/s to 23.77 MB/s, an increase of about 376%.

These considerable gains indicate that, in the reference version, the bottleneck was not solely the network, but a combination of CPU-intensive operations and communication. The significant reduction in computation time due to AVX2 optimizations allowed the system to process cryptographic operations much faster, thereby freeing up the communication pipeline and enabling much higher throughputs that were likely limited by the speed of the cryptographic computations themselves in the reference version.

The throughput symmetry between Rank 0 and Rank 1 is maintained, which is a good indication of the robustness of the test environment and the implementation. These results demonstrate that AVX2 optimizations are essential for fully leveraging network capabilities in applications where post-quantum cryptographic operations are integrated into communications.

9. CPU Profiling Analysis with `perf`

To identify the main computational bottlenecks in Kyber, a detailed CPU profiling was conducted using the Linux tool `perf`. This analysis highlights the most CPU-intensive functions and guides optimization efforts for HPC environments.

Table 18
Main CPU Hotspots by Parameterization (overhead %)

Function	Kyber-512	Kyber-768	Kyber-1024
KeccakF1600_StatePermute	20.99	27.03	25.81
montgomery_reduce	19.01	19.05	19.49
invntt	11.40	5.34	7.11
ntt	8.95	12.46	8.81
barrett_reduce	8.79	7.52	6.00
basemul	7.57	6.47	8.38
Total crypto core	**76.71**	**77.87**	**75.60**

Table 19
Estimated Memory Footprint by Parameterization

Parameterization	Memory/operation (bytes)	Memory/operation (kB)	1000 operations (MB)
Kyber-512	3232	3.16	3.08
Kyber-768	4704	4.59	4.49
Kyber-1024	6336	6.19	6.04

9.1. Profiling Methodology

Profiling was performed with `perf record` on the speed benchmarks (`test_speed512`, `test_speed768`, `test_speed1024`) for each Kyber parameterization. Measurements focused on CPU cycles, with call trace recording for fine-grained function analysis.

9.2. CPU Hotspot Analysis

The results show a strong concentration of CPU usage on a small number of main functions, consistently across all parameterizations. Table 18 summarizes the most resource-consuming functions.

9.3. Learnings and Recommendations

- **Keccak (SHAKE128/256)** is the primary bottleneck (21–27% of CPU time), used for deterministic key generation, encapsulation, and internal hashing functions.
- **Montgomery and Barrett reduction** together account for nearly 28% of CPU time, essential for modular arithmetic in polynomial operations.
- **NTT and invNTT** (forward and inverse number theoretic transform) total 15–20% of CPU time, with a relative cost that decreases for larger parameterizations due to better cache utilization.
- **Priority Optimization**: Accelerating Keccak (via vectorization, hardware SHA-3 instructions, or GPU offload) and modular arithmetic (SIMD, assembly) would yield the most significant gain.
- **Algorithmic Stability**: The cost distribution remains stable regardless of the security level, which facilitates large-scale optimization.

9.4. Estimated Memory Footprint

Table 19 summarizes the estimated memory footprint for a complete KEM operation (public key, secret key, ciphertext, shared secret).

In summary, CPU profiling shows that Kyber's performance primarily relies on the efficiency of hashing functions and modular arithmetic. Optimizations targeting these critical points are a priority for HPC deployments.

10. Performance Summary and Recommendations

This section provides a synthesis of the experimental results and highlights key trends for deploying Kyber in HPC environments.

10.1. Overall Performance Summary

The measurements performed show:

- **Latency (Average RTT, 1000 iterations)**:
  - Kyber-512: 319 μs
  - Kyber-768: 434 μs
  - Kyber-1024: 599 μs
- **Effective Network Throughput**: stable around 4.8–5.0 MB/s, regardless of parameter size.
- **Network Transfer**: public key transfer time remains negligible (<1 μs), while ciphertext transfer becomes significant for higher parameters.
- **Cryptographic Operation Overhead**: almost all of the total latency is due to cryptographic computations (key generation, encapsulation, decapsulation).
- **Cryptographic Quality**: perfect validation of test vectors, entropy greater than 97% of the theoretical value, full compliance with NIST PQC specifications.

10.2. Trends and Practical Implications

Scalability and Overhead

- Total latency increases predictably with the security level: +36% between Kyber-512 and Kyber-768, +38% between Kyber-768 and Kyber-1024.
- Network throughput remains stable despite doubling the data volume between Kyber-512 and Kyber-1024, demonstrating the robustness of the HPC infrastructure to PQC.
- Memory footprint increases linearly with the security level (3.2 kB to 6.3 kB per KEM operation).

Identified Bottlenecks

- Cryptographic operations dominate (95%+ of total time).
- Ciphertext transfer becomes a non-negligible factor for higher security levels.
- Some extreme latency values (observed in max RTTs) can be attributed to system factors, such as task scheduling or memory management, and should be considered for latency-sensitive applications.

10.3. Recommendations for HPC Deployment

- **Kyber-512 (NIST Level 1)**: To be preferred for HPC applications where performance is paramount and moderate security is sufficient. Very low latency suitable for frequent exchanges.
- **Kyber-768 (NIST Level 3)**: Optimal security/performance compromise for most scientific and industrial uses.
- **Kyber-1024 (NIST Level 5)**: Reserved for use cases requiring maximum security, at the cost of increased latency and memory.

Practical Advice

- Current HPC network infrastructure easily supports the data volumes generated by Kyber, even at the highest security levels.
- Consider the linear growth of memory for massive deployments (thousands of simultaneous connections).
- Budget 300 to 600 μs per key establishment, depending on the security level.
- To minimize latency outliers, optimize system management and process scheduling.

Conclusion

This study rigorously evaluated and profiled the performance of the Kyber post-quantum key exchange algorithm in a High-Performance Computing (HPC) environment, focusing on both reference and optimized (AVX2) implementations across its different security levels (Kyber-512, Kyber-768, Kyber-1024). Our in-depth analyses of metrics such as latency, bandwidth, and CPU footprint clearly demonstrated the inherent trade-offs between security and cryptographic performance.

The obtained results highlight the critical importance of hardware optimizations, particularly the integration of AVX2 instructions, which yielded substantial performance gains, significantly reducing KEM operation latency and increasing effective throughput. This improvement is crucial for the adoption of post-quantum cryptography in demanding infrastructures like supercomputers, where efficiency is paramount. By identifying major CPU "hotspots," we provide concrete avenues for future specific optimizations.

Ultimately, this work offers a detailed characterization of Kyber's behavior under real HPC conditions, filling an important gap in the literature. The information and recommendations stemming from this research are essential to guide system architects and developers in the transition to a post-quantum cryptographic era, ensuring both security against emerging quantum threats and the preservation of HPC system operational efficiency.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] N. I. of Standards, Technology, Post-quantum cryptography standardization, https://csrc.nist.gov/Projects/post-quantum-cryptography, 2024. Accessed: 2025-07-11.

[2] J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, D. Stehl'e, Crystals–kyber: a cca-secure module-lattice-based kem, 2018 IEEE European Symposium on Security and Privacy (EuroS&P) (2018) 353–367. doi:10.1109/EuroSP.2018.00031.

[3] S. Hueber, D. K"ugler, J. Buchmann, P. Krausz, I. von Maurich, T. P"oppelmann, W. Schindler, Performance evaluation of post-quantum cryptography in high-performance computing environments, Future Generation Computer Systems 143 (2023) 1–13. doi:10.1016/j.future.2023.01.008.

[4] N. Bindel, J. Brendel, M. Fischlin, B. Goncalves, D. Stebila, The performance of post-quantum tls 1.3, ACM Transactions on Privacy and Security (TOPS) 25 (2022) 1–36. doi:10.1145/3546065.

[5] G. Gong, S. Hao, X. Yang, L. Wang, Optimizing post-quantum cryptography on intel processors: The case of crystals-kyber, Computers & Security 108 (2021) 102377. doi:10.1016/j.cose.2021.102377.

[6] S. Chen, J. Zhang, Z. Gao, Benchmarking post-quantum cryptography in tls: A performance evaluation on various platforms, in: 2021 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE, 2021, pp. 1588–1595. doi:10.1109/TrustCom53375.2021.00224.