Evaluating Post-Quantum Cryptographic Algorithms on Resource-Constrained Devices

Jesus Lopez[†]
Department of Computer Science
University of Texas at El Paso
jlopez126@miners.utep.edu

Viviana Cadena[†]
Department of Computer Science
University of Texas at El Paso
vcadena1@miners.utep.edu

Mohammad Saidur Rahman
Department of Computer Science
University of Texas at El Paso
msrahman3@utep.edu

***Abstract*—The rapid advancement of quantum computing poses a critical threat to classical cryptographic algorithms such as RSA and ECC, particularly in Internet of Things (IoT) devices, where secure communication is essential but often constrained by limited computational resources. This paper investigates the feasibility of deploying post-quantum cryptography (PQC) algorithms on resource-constrained devices. In particular, we implement three PQC algorithms — BIKE, CRYSTALS-Kyber, and HQC — on a lightweight IoT platform built with Raspberry Pi devices. Leveraging the Open Quantum Safe (`liboqs`) library in conjunction with `mbedTLS`, we develop quantum-secure key exchange protocols, and evaluate their performance in terms of computational overhead, memory usage, and energy consumption for quantum secure communication. Experimental results demonstrate that the integration of PQC algorithms on constrained hardware is practical, reinforcing the urgent need for quantum-resilient cryptographic frameworks in next-generation IoT devices. The implementation of this paper is available at https://iqsec-lab.github.io/PQC-IoT/.**

***Index Terms*—Post-Quantum Cryptography; Constrained Devices; IoT Security; Quantum-Secure Communication**

I. Introduction

The widespread integration of Internet of Things (IoT) devices into modern infrastructure, from home automation systems to connected vehicles, have introduced new levels of connectivity, convenience, and efficiency. These devices, typically powered by low-cost microcontrollers, operate under stringent resource constraints, including limited processing power, memory, and energy capacity [1]. While this lightweight design supports scalability, it often comes at the cost of robust security [2], [3]. As a result, many consumer-grade IoT devices lack strong cryptographic protections, leaving personal data and network integrity vulnerable to attack.

Securing IoT systems has long been a priority, but the rapid advancement of quantum computing elevates the urgency of this challenge [4]. Classical public-key cryptographic systems such as RSA and elliptic-curve cryptography (ECC), which secure much of today's digital communication including IoT applications, are vulnerable to quantum attacks [5]. In particular, Shor's algorithm can efficiently solve the integer factorization and discrete logarithm problems that underpin RSA and ECC, rendering them insecure in the presence of large-scale quantum computers [6], [7]. This looming threat necessitates the adoption of cryptographic algorithms that remain secure even against quantum adversaries.

Post-quantum cryptography (PQC) addresses this need by developing cryptographic schemes based on mathematical complexity and are resistant to quantum-powered attacks [7], [8]. In recognition of the urgent need for standardization, the U.S. National Institute of Standards and Technology (NIST) launched a multi-year effort to identify quantum-resistant cryptographic algorithms [9], [10]. In July 2022, NIST announced the first set of finalists: CRYSTALS-Kyber for key encapsulation, and CRYSTALS-Dilithium, Falcon, and SPHINCS+ for digital signatures [10].

In practice, secure communication in embedded systems often relies on Transport Layer Security (TLS) or TLS-inspired protocols, where key encapsulation mechanisms (KEM's) play a central role in negotiating session keys [11]. Improving the classical key exchange components in TLS with PQC-based KEM's is one of the most immediate and critical steps in transitioning toward quantum-resistant infrastructure [12]. However, most studies on PQC performance have focused on general-purpose processors, leaving a gap in understanding their behavior in real-world embedded environments [10].

In this paper, we conduct a systematic evaluation of three NIST-designated PQC key encapsulation mechanisms (KEM) — Bit Flipping Key Encapsulation (BIKE) [13], Hamming Quasi-Cyclic (HQC) [14], and CRYSTALS-Kyber [15] on Raspberry Pi-based platforms representative of embedded IoT devices. We measure and compare their performance across four critical dimensions – execution time, power consumption, memory usage, and device temperature. The goal is to assess the feasibility and efficiency of PQC algorithms for secure communication in resource-constrained environments, and to provide practical insights for future PQC integration in embedded systems.

In summary, the contributions of this paper are as follows:

- We present a standardized, reproducible benchmarking methodology for evaluating PQC algorithms using a TLS-inspired client-server model deployed on constrained embedded platforms.
- We implement and evaluate three NIST-designated PQC KEM's – BIKE, HQC, and CRYSTALS-Kyber, at multiple security levels, measuring execution time, power

[†]Equal contribution.

consumption, memory usage, and thermal behavior on Raspberry Pi-based systems.
- Our findings show that CRYSTALS-Kyber provides the most favorable trade-off across performance metrics, demonstrating high suitability for secure deployment in constrained IoT environments.
- We identify key trade-offs in BIKE and HQC, showing that while BIKE achieves the lowest memory usage, it incurs significant latency and power costs at higher security levels; HQC offers moderate runtime efficiency but suffers from high memory demand and thermal overhead.

II. Related Work

Research on securing Internet of Things (IoT) devices with post-quantum cryptography (PQC) algorithms has received significant attention from industry and academia due to the growing risks posed by quantum computing [16]–[18]. Prior work suggests that public-key schemes such as RSA, DSA, Diffie–Hellman, and ECC can be broken by Shor's algorithm by efficiently solving integer factorization and discrete logarithm problems [6], [19], [20]. Furthermore, Grover's algorithm reduces the security of symmetric-key encryption (e.g., AES) and hash-based message authentication codes (e.g., GMAC, Poly1305) by providing a quadratic speedup [7], [21], [22]. However, symmetric key systems can still be secured by increasing the key sizes, but public-key systems require new designs.

Prior work also emphasized the quantum threats to Internet-of-Things (IoT) devices focusing on threat models, resource usage, models and security analyses for IoT applications [23]–[25]. Liu et al. [26] identified key challenges in integrating post-quantum cryptography (PQC) into IoT, notably limited memory, energy constraints, and the computational overhead of large key sizes, particularly for lattice- and code-based schemes. They conducted a comprehensive survey of over 30 studies on PQC performance, optimization, and GPU acceleration for resource-constrained devices, highlighting Kyber and Dilithium as promising candidates due to their balance between speed and resource consumption. Their work also established benchmark metrics—including speed, code size, transmission size, memory usage, energy, and power—and emphasized the need for a unified evaluation methodology aligned with NIST standardization efforts to ensure consistent benchmarking across constrained IoT platforms.

Recent benchmarking of NIST-selected post-quantum cryptographic (PQC) algorithms on Raspberry Pi 4 devices [27] evaluated Round 4 winners across lattice-, hash-, code-, and isogeny-based categories, excluding SIKE. The study assessed Kyber, Dilithium, Falcon, SPHINCS+, BIKE, Classic McEliece, and HQC across key generation, encapsulation, signing, verification, and TLS 1.3 handshake performance, showing that Kyber-768 and Dilithium3 achieve superior key exchange and signing efficiency, respectively, while Falcon yields the smallest handshake size. Despite these gains, the need for standardized optimization practices was emphasized [27]. Further optimization efforts have targeted code-based schemes; for instance, an optimized HQC implementation on an ARM Cortex-M4 platform using ChibiOS RTOS [28] achieved 96% and 95% reductions in key generation and encapsulation times, respectively, while lowering the memory footprint compared to PQClean. However, further improvements and side-channel resistance evaluations remain necessary [28]. In parallel, high-level synthesis (HLS) techniques have been explored for hardware acceleration of NIST Round 3 candidates [29], yielding up to 43× and 18× latency reductions for AES and NTT primitives, respectively. Integrating these optimizations into CRYSTALS-Dilithium and CRYSTALS-Kyber resulted in approximately 40% area savings and 10% latency improvements, demonstrating that Dilithium2 architectures outperform Dilithium-medium and that HLS provides an efficient alternative to hand-coded RTL for PQC design.

III. Preliminaries

To ensure a fair comparison among algorithms, we evaluate only the Key Encapsulation Mechanism (KEM) component. Symmetric encryption of plaintext is performed using the Advanced Encryption Standard in Galois/Counter Mode (AES-GCM), and SHA-256 is employed for key derivation.

A. AES-GCM

AES-GCM is a widely adopted authenticated encryption algorithm that provides both confidentiality and integrity. It integrates the AES block cipher in Counter (CTR) mode with Galois field multiplication (GHASH) to generate an authentication tag. GHASH compresses the Additional Authenticated Data (AAD) and ciphertext into a single block, ensuring tamper detection [30], [31].

AES-GCM is highly parallelizable, FIPS-approved, and deployed in protocols such as TLS 1.3, IPsec, and SSH [32], [33]. It performs two operations: authenticated encryption and authenticated decryption.

During authenticated encryption, the algorithm requires:

- Secret key: Symmetric encryption key.
- Initialization vector (IV): Typically a unique 96-bit nonce.
- Plaintext: Data to encrypt.
- Optional AAD: Authenticated but unencrypted metadata.

The outputs are:

- Ciphertext: The encrypted message.
- Authentication tag: A fixed-length tag, typically 128 bits.

During authenticated decryption, the ciphertext, IV, authentication tag, and optional AAD are used to verify authenticity. Decryption proceeds only if the recalculated tag matches the received tag; otherwise, an error is returned [30].

Proper IV management is critical: IV reuse under the same key breaks security guarantees [34]. In this work, IVs are randomized for every encryption operation to mitigate this risk.

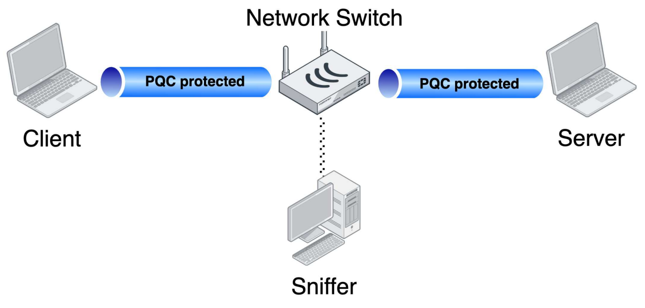


Fig. 1. Experimental setup for PQC-protected communication between a client and a server. A sniffer is connected through a network switch to monitor traffic flow and passively verify correct message routing.

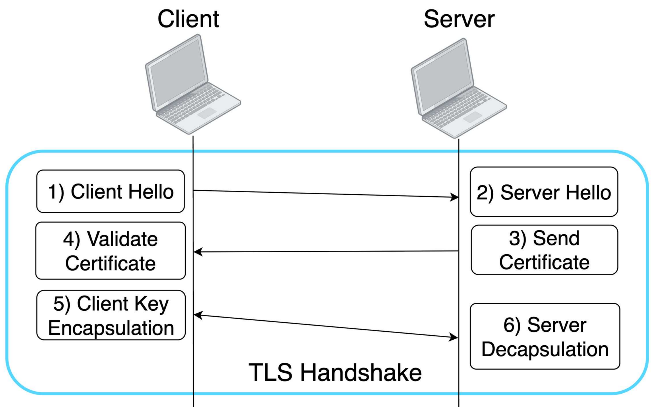


Fig. 2. Simplified TLS handshake protocol, illustrating the exchange between the client and the server.

B. SHA-256

SHA-256, standardized in FIPS-180-4, is a cryptographic hash function that produces a 256-bit output from input data of arbitrary length [35]. It operates by dividing the input into 512-bit blocks and applying a compression function based on logical operations and modular addition. Designed to offer collision and preimage resistance, SHA-256 also ensures that small changes in input lead to significantly different outputs. These properties make it a reliable choice for deriving symmetric keys from shared secrets in key encapsulation mechanisms (KEMs). Due to its balance of computational efficiency and security, SHA-256 remains well-suited for use in constrained environments such as IoT systems.

C. Naming Conventions

The National Institute of Standards and Technology (NIST) recommends the following naming conventions for clarity regarding the algorithms discussed in this article: BIKE [13], HQC [14], and CRYSTALS-Kyber [15]. Specifically, CRYSTALS-Kyber may also be referred to as Federal Information Processing Standards (FIPS) Publication 203 [36], as it has been officially standardized. BIKE can only be referenced by its algorithm name, since it is a finalist in the NIST PQC competition but is not selected for standardization. Finally, HQC may be temporarily referred to by its algorithm name, as it has been selected for standardization, but its corresponding FIPS publication has not yet been released at the time of this work [37].

IV. Methodology

This study evaluates the performance of three post-quantum key encapsulation mechanisms (KEMs) – BIKE [13], CRYSTALS-Kyber [38], and HQC [14], within a simulated client-server communication framework. The goal is to assess the feasibility of deploying these algorithms on constrained embedded systems by measuring their power consumption, execution time, memory usage, and communication throughput.

A. Experimental Setup

The experimental setup, shown in Figure 1, consisted of a Raspberry Pi 3 Model B+ as the client and a Raspberry Pi 5 as the server. Both devices ran Raspberry Pi OS Lite, a Debian-based system selected for its minimal overhead and lightweight design. An offline network switch connected the two devices, providing an isolated environment free from external interference to ensure consistent and reproducible measurements.

All software is developed in C, using mbedTLS (version 3.6) as the primary cryptographic library [39], extended with liboqs from the Open Quantum Safe (OQS) project (version 0.12) [40]. Each KEM is integrated independently into the communication protocol to allow for a fair evaluation.

Testing includes three parameter sets for each KEM to assess performance at varying security levels. BIKE is evaluated with BIKE-L1, BIKE-L3, and BIKE-L5; HQC with HQC-128, HQC-192, and HQC-256; and CRYSTALS-Kyber with Kyber512, Kyber768, and Kyber1024. Each one of the levels align with the NIST PQC standardization goals [41] of providing 128-bit, 192-bit and 256-bit key. These levels mirror classical cryptographic strength AES-128, AES-192/SHA-384, AES-256/SHA-512. This selection enables a consistent comparison of the algorithm families across different cryptographic strengths.

A passive network sniffer, connected to the switch, monitors client-server communications. Packet exchanges are captured using Wireshark over a wired interface [42], ensuring accurate observation without introducing overhead or altering the communication behavior.

B. Implementation Procedure

A custom lightweight protocol is implemented for benchmarking, consisting of the following steps:

- Handshake: The client initiates a secure session by establishing a TCP connection with the server, shown in Figure 2.
- Key Exchange:
  - The server generates a key pair using the selected post-quantum key encapsulation mechanism (PQC-KEM) and sends the public key to the client.
  - The client performs key encapsulation and returns the resulting ciphertext and shared secret to the server.

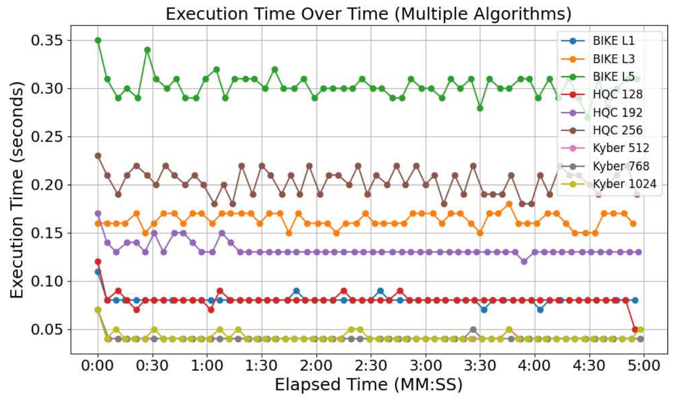


Fig. 3. Execution time comparison of post-quantum key encapsulation mechanisms (KEMs) over a five-minute interval. The three levels of BIKE, HQC, and Kyber algorithms were evaluated across multiple security levels.

- Session Key Derivation: Both the client and server derive a common session key from the KEM output.
- Secure Message Exchange: To validate successful key agreement, the client encrypts a message using AES-GCM and transmits it to the server.

C. Performance Evaluation Procedure

To evaluate the performance of each algorithm, the client and server applications are executed concurrently in isolated sessions. Four text files of varying sizes (208 bytes, 731 bytes, 1235 bytes, and 2328 bytes) are used to simulate different communication loads and observe the impact of message size on system performance. The use of text files were chosen for their simplicity and transparency, allowing us to easily control the data size, format, and structure. This approach is straightforward for manual testing, reproducibility and consistency. Each session runs for a fixed duration of five minutes, followed by a five-minute rest period to allow the devices to return to baseline thermal and performance conditions.

Performance data is collected continuously during each session. The following metrics are recorded for analysis:

- Total execution time (seconds)
- Power consumption (watts)
- Memory usage (kilobytes)
- Device temperature (degrees Celsius)

In the case of power consumption, we are measuring the server and the client under low-load conditions, which means that the system is idle mainly or under minimal stress. In contrast, High-load conditions represent when the system is under stress; we used this to measure the peak power consumption of a typical use in a real-world application.

V. Evaluation

A. Results

This section presents the performance evaluation of three post-quantum key encapsulation mechanisms (KEMs) – BIKE, HQC, and Kyber – across different security levels.

TABLE I
Execution Time (in seconds) for Post-Quantum Cryptography Algorithms.

PQC Algorithm	Execution Time
BIKE L1	0.081 ± 0.005
BIKE L3	0.164 ± 0.007
BIKE L5	0.302 ± 0.013
HQC 128	0.081 ± 0.007
HQC 192	0.133 ± 0.008
HQC 256	0.203 ± 0.013
Kyber 512	0.041 ± 0.004
Kyber 768	0.041 ± 0.004
Kyber 1024	0.042 ± 0.005

TABLE II
Power consumption (Watts) of PQC Algorithms.

PQC Algorithm	Low		High	
	Client	Server	Client	Server
BIKE L1	3.2	3.5	3.5	4.7
BIKE L3	3.2	3.5	3.5	4.8
BIKE L5	3.2	3.5	3.5	5.2
HQC 128	3.2	3.4	3.6	4.0
HQC 192	3.2	3.5	3.6	4.1
HQC 256	3.2	3.5	3.7	4.7
Kyber 512	3.2	3.4	3.5	3.9
Kyber 768	3.2	3.5	3.5	3.9
Kyber 1024	3.2	3.5	3.5	3.9

1) Execution Time: The execution time of each algorithm is evaluated over continuous five-minute sessions, with metrics recorded at each iteration as described in subsection IV-C.

Figure 3 shows the execution time behavior of all evaluated algorithms. Among the families tested, the Kyber variants consistently exhibit the lowest and most stable execution times. Kyber-512, Kyber-768, and Kyber-1024 report mean execution times of 0.041 ± 0.004 seconds, 0.041 ± 0.004 seconds, and 0.042 ± 0.005 seconds, respectively, as summarized in Table I. The small observed fluctuations, with standard deviations between ±0.004 and ±0.005 seconds, indicate that Kyber maintains a highly consistent runtime, with minimal sensitivity to external conditions or input variations.

In contrast, the BIKE family shows significantly higher execution times. BIKE-L1 and BIKE-L3 report mean execution times of 0.081 ± 0.005 seconds and 0.164 ± 0.007 seconds, respectively, while BIKE-L5 achieves the highest latency at 0.302 ± 0.013 seconds.

The HQC family demonstrates intermediate performance. HQC-128 matches BIKE-L1 with an execution time of 0.081 ± 0.007 seconds, while HQC-192 and HQC-256 report 0.133 ± 0.008 seconds and 0.203 ± 0.013 seconds, respectively. The higher standard deviations observed for BIKE-L5 and HQC-256 (±0.013 seconds) reflect greater variability compared to Kyber variants.

2) Power Consumption: Power consumption is measured separately under low-load and high-load conditions for both the client and the server,as shown in Table II. Low-load corresponds to idle and High-load simulates peak usage scenarios representative of real-world applications.

Under low-load conditions, all algorithms demonstrate sim-

TABLE III
MEMORY USAGE (IN KILOBYTES) OF PQC ALGORITHMS IN FOUR NETWORK SCENARIOS.

Algorithm	**networkSim1**	**networkSim2**	**networkSim3**	**networkSim4**
BIKE L1	5632.00 ± 0.00	5632.00 ± 0.00	5632.00 ± 0.00	5629.79 ± 16.81
BIKE L3	5627.59 ± 33.61	5629.79 ± 16.81	5620.97 ± 43.45	5629.79± 16.81
BIKE L5	5851.43 ± 93.63	5888.00 ± 0.00	5858.29 ± 80.92	5858.29 ± 84.52
HQC 128	5757.83 ± 16.66	5760.00 ± 0.00	5749.15 ± 35.95	5753.49 ± 28.36
HQC 192	5998.34 ± 55.97	5987.31 ± 68.00	6000.55 ± 48.42	6000.55 ± 59.11
HQC 256	5874.53 ± 46.43	5881.26 ± 28.84	5879.02 ± 32.99	5883.51 ± 23.76
Kyber 512	5888.00 ± 0.00	5885.83 ± 16.66	5888.00 ± 0.00	5760.00 ± 0.00
Kyber 768	5760.00 ± 0.00	5757.83 ± 16.66	5757.83 ± 16.66	5885.83 ± 16.66
Kyber 1024	5888.00 ± 0.00	5888.00 ± 0.00	5888.00 ± 0.00	5883.66 ± 33.33

TABLE IV
MEAN AND STANDARD DEVIATION OF TEMPERATURES ON SERVER AND CLIENT DEVICES (IN °C)

Algorithm	**Server Temperature (°C)**				**Client Temperature (°C)**			
	networkSim1	networkSim2	networkSim3	networkSim4	networkSim1	networkSim2	networkSim3	networkSim4
BIKE L1	48.37±0.88	50.55±0.95	50.83±0.93	51.86±1.02	48.88 ±0.59	49.77±0.42	50.45±0.42	52.64±0.48
BIKE L3	51.04±1.32	51.33±1.09	51.11±1.27	52.26±1.09	52.69±0.36	52.60±0.48	52.39±0.46	52.26±0.42
BIKE L5	51.53±1.29	51.75±1.30	52.04±1.26	51.84±1.09	53.41±0.47	53.40±0.49	53.38±0.51	53.51±0.38
HQC 128	48.23±0.88	49.16±1.03	49.49±0.90	51.70±0.74	53.29±0.40	53.09±0.55	53.48±0.49	53.66±0.42
HQC 192	48.43±0.95	49.05±0.90	50.81±1.21	51.31±1.08	53.86±0.46	54.20±0.52	53.92±0.48	54.08±0.45
HQC 256	50.22±0.85	50.07±1.01	50.53±1.08	51.11±0.90	53.90±0.48	52.21±0.64	52.41±0.60	52.44±0.46
Kyber 512	47.96±0.87	47.54±1.29	49.03±1.20	50.01±1.16	52.15±0.44	52.15±0.53	51.98±0.48	52.35±0.49
Kyber 768	47.27±0.70	47.62±0.97	48.92±1.06	49.66±1.16	52.09±0.47	52.14±0.54	52.20±0.48	51.97±0.46
Kyber 1024	47.85±1.06	47.62±1.19	49.27±1.07	49.81±1.08	52.21±0.46	52.24±0.40	51.95±0.47	52.06±0.48

ilar behavior, consuming approximately 3.2 Watts on the client side. Server-side consumption varies slightly between 3.4 to 3.5 Watts across all algorithms. Under high-load conditions, differences become more pronounced. BIKE-L5 records the highest server-side power consumption at 5.2 Watts, followed by BIKE-L3 at 4.8 Watts and BIKE-L1 at 4.7 Watts. On the client side, all BIKE variants maintain 3.5 Watts.

The HQC family shows moderate high-load power consumption. HQC-128 and HQC-192 consume 4.0 Watts and 4.1 Watts on the server side, respectively, while HQC-256 rises slightly to 4.7 Watts. Client-side power consumption for HQC variants ranges from 3.6 to 3.7 Watts.

The Kyber family maintains the lowest power consumption under high-load conditions, with Kyber-512, Kyber-768, and Kyber-1024 each consuming approximately 3.5 Watts on the client and 3.9 Watts on the server.

3) Memory Usage: Memory usage for each algorithm is summarized in Table III, measured across four different communication scenarios (networkSim1 to networkSim4) corresponding to increasing file sizes. The four input files contain increasingly larger text-based inputs, with sizes of 208 bytes, 731 bytes, 1235 bytes, and 2328 bytes, respectively. This setup allows us to evaluate how our measurements changes with the change of input size, simulating varying communication loads and scenarios.

In networkSim1 (208 bytes), BIKE-L1 records a mean memory usage of 5632.00 ± 0.00 KB, while BIKE-L3 uses 5627.59 ± 33.61 KB. These are the lowest among all tested algorithms. In contrast, HQC-192 and Kyber-512 reach the highest memory usage, reporting 5998.34 ± 55.97 KB and 5888.00 ± 0.00 KB, respectively. Kyber-1024 similarly reports 5888.00 ± 0.00 KB.

As the file size increases in networkSim2 (731 bytes), networkSim3 (1235 bytes), and networkSim4 (2328 bytes), overall memory usage remains relatively stable for most algorithms. HQC-192 consistently shows the highest memory usage, ranging between 5987.31 ± 68.00 KB and 6000.55 ± 59.11 KB across the four simulations. BIKE-L1 and BIKE-L3 continue to demonstrate the lowest memory usage, maintaining values between 5620.97 ± 43.45 KB and 5632.00 ± 0.00 KB.

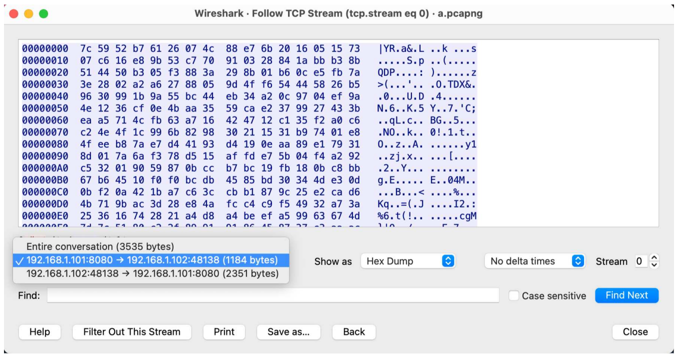
Variability in memory usage is reflected in the standard deviation values. BIKE-L5 displays the greatest fluctuation, with deviations reaching up to ± 93.63 KB in networkSim1. In comparison, Kyber-512 and Kyber-1024 show no measurable fluctuation in several simulations, indicating highly stable memory consumption.

Overall, BIKE-L1, BIKE-L3, and Kyber-512 demonstrate both low and stable memory usage, while HQC-192 shows higher and more variable memory demand across all file sizes.

4) Device Temperature: Temperature measurements for both client and server devices are reported in Table IV.

For server temperatures, the Kyber family consistently records the lowest operating temperatures across all scenarios. Kyber-768 maintains the lowest average temperatures, ranging from 47.27°C to 49.66°C across the four network simulations. Kyber-512 and Kyber-1024 report similar thermal profiles, with temperatures between 47.54°C and 50.01°C.

The BIKE and HQC families demonstrate slightly higher operating temperatures. BIKE-L1, BIKE-L3, and BIKE-L5 show average server temperatures exceeding 50°C under load, with BIKE-L5 reaching up to 52.04°C in networkSim3. Similarly, HQC-256 records temperatures ranging from 50.07°C



```
00000000  7c 59 52 b7 61 26 07 4c  88 e7 6b 20 16 05 15 73   |YR.a&.L ..k ...s
00000010  07 c6 16 e8 9b 53 c7 70  91 03 28 84 1a bb b3 8b   .....S.p ..(.....
00000020  51 44 50 b3 05 f3 88 3a  29 8b 01 b6 0c e5 fb 7a   QDP....: )......z
00000030  3e 28 02 a2 a6 27 88 05  9d 4f f6 54 44 58 26 b5   >(...'.. .O.TDX&.
00000040  96 30 99 1b 9a 55 bc 44  eb 34 a2 0c 97 04 ef 9a   .0...U.D .4......
00000050  4e 12 36 cf 0e 4b aa 35  59 ca e2 37 99 27 43 3b   N.6..K.5 Y..7.'C;
00000060  ea a5 71 4c fb 63 a7 16  42 47 12 c1 35 f2 a0 c6   ..qL.c.. BG..5...
00000070  c2 4e 4f 1c 99 6b 82 98  30 21 15 31 b9 74 01 e8   .NO..k.. 0!.1.t..
00000080  4f ee b8 7a e7 d4 41 93  d4 19 0e aa 89 e1 79 31   O..z..A. ......y1
00000090  8d 01 7a 6a f3 78 d5 15  af fd e7 5b 04 f4 a2 92   ..zj.x.. ...[....
000000A0  c5 32 01 90 59 87 0b cc  b7 bc 19 fb 18 0b c8 bb   .2..Y... ........
000000B0  67 b6 45 10 f0 f0 bc db  45 85 bd 30 34 4d e3 0d   g.E..... E..04M..
000000C0  0b f2 0a 42 1b a7 c6 3c  cb b1 87 9c 25 e2 ca d6   ...B...< ....%...
000000D0  4b 71 9b ac 3d 28 e8 4a  fc c4 c9 f5 49 32 a7 3a   Kq..=(.J ....I2.:
000000E0  25 36 16 74 28 21 a4 d8  a4 be ef a5 99 63 67 4d   %6.t(!.. .....cgM
```

Entire conversation (3535 bytes)
✓ 192.168.1.101:8080 → 192.168.1.102:48138 (1184 bytes)
192.168.1.102:48138 → 192.168.1.101:8080 (2351 bytes)

Show as: Hex Dump No delta times Stream 0

Fig. 4. Wireshark output showing the captured TCP stream. The highlighted packet is 1,184 bytes in size, which corresponds to the expected ciphertext length produced by the Kyber-768 PQC algorithm.

to 51.11°C across different simulations.

Client-side temperatures follow a similar trend. Kyber variants again maintain lower thermal profiles compared to BIKE and HQC. Some missing data points are observed in the client-side measurements for BIKE-L3 and BIKE-L5, likely due to inconsistencies during data collection. Nonetheless, the overall trend aligns with server-side observations, highlighting Kyber's thermal efficiency relative to BIKE and HQC.

5) *Packet Sniffer:* A sniffer was placed between Alice and Bob, as illustrated in Figure 1, to monitor the communication channel of client and server. ARP spoofing is employed to redirect and capture network packets. Since PQC protocols are not yet integrated into mainstream protocols such as TLS, Wireshark is currently unable to natively recognize them. Given that our implementation uses PQC within a custom protocol, protocol-specific identification is not feasible using standard packet analyzers. Therefore, analysis is performed by directly inspecting the contents of the transmitted packets.

Figure 4 depicts, as an example, the use of the Kyber-768 algorithm to demonstrate a specific packet in the network trace. The size of the transmitted message is 1,184 bytes, which corresponds to the expected ciphertext length generated by the Kyber-768 key encapsulation mechanism.

B. Analysis

The evaluation results highlight significant performance differences across the three post-quantum key encapsulation mechanisms (KEMs): BIKE, HQC, and Kyber.

Across all metrics, Kyber consistently outperforms the other algorithms. Kyber variants achieve the lowest execution times, with Kyber-512 and Kyber-768 operating at 0.041 ± 0.004 seconds and Kyber-1024 at 0.042 ± 0.005 seconds. They also demonstrate minimal runtime variability. Correspondingly, Kyber maintains the lowest high-load power consumption, approximately 3.5 Watts on the client side and 3.9 Watts on the server side, and the lowest operating temperatures, ranging between 47.27°C and 50.01°C. Memory usage for Kyber variants is slightly higher than BIKE but remains highly stable, with almost no fluctuation in several scenarios. These results indicate that Kyber offers a balanced profile of computational efficiency, low energy consumption, and thermal stability, making it highly suitable for constrained embedded systems.

BIKE shows mixed performance. While BIKE-L1 and BIKE-L3 maintain relatively low memory footprints (around 5630 KB) and low idle power consumption, their execution times are significantly higher compared to Kyber. BIKE-L5, in particular, exhibits the highest latency (0.302 ± 0.013 seconds) and the highest server-side power consumption under load (5.2 Watts). BIKE variants also show higher thermal profiles, with BIKE-L5 reaching up to 52.04°C, and increased variability in memory usage, particularly for BIKE-L5. These trends suggest that while BIKE may be competitive at lower security levels, its scalability to higher security parameters introduces notable performance penalties.

HQC demonstrates intermediate behavior across most metrics. HQC-128 offers execution times comparable to BIKE-L1 (0.081 ± 0.007 seconds), but HQC-192 and HQC-256 show increases in memory consumption (up to 6000.55 ± 59.11 KB) and power consumption (server-side power reaching 4.7 Watts for HQC-256). HQC variants also experience elevated operating temperatures, similar to BIKE. The increased memory demand and variability, especially at higher security levels, may pose challenges for resource-constrained platforms.

Overall, the Kyber family provides the most balanced and efficient performance across all tested metrics. BIKE and HQC offer viable alternatives depending on application-specific constraints but exhibit trade-offs in execution time, energy consumption, and memory stability, particularly at higher security levels.

VI. Discussion

While post-quantum cryptography offers strong security guarantees, its deployment on resource-constrained devices remains non-trivial. Our results reveal that CRYSTALS-Kyber, a lattice-based scheme built on the Module Learning with Errors (M-LWE) problem [15], consistently outperforms code-based alternatives BIKE and HQC in execution time, energy efficiency, and thermal stability. Kyber's computational advantages and implementation maturity [43] make it particularly well-suited for embedded environments.

Despite optimization efforts, HQC and BIKE exhibit higher resource demands than Kyber. Hardware-accelerated designs for HQC [28] and constant-time implementations of BIKE for embedded platforms [44] improve performance but do not fully address their elevated energy and time costs. Our results confirm these trends: both schemes show increased energy consumption and memory usage, particularly at higher security levels, limiting their suitability for constrained environments.

All three algorithms maintain acceptable power usage on the client side, but BIKE-L1 and BIKE-L3 are preferable for systems with strict memory constraints, consistently demonstrating the lowest memory consumption across all test scenarios. To ensure fair comparison, all experiments are conducted under consistent conditions using identical operating systems (Raspberry Pi OS Lite v6.6), codebases differing only by the integrated KEM, and an isolated network to eliminate external interference. A Raspberry Pi 5 functions as the server, and a Raspberry Pi 3 as the client, modeling practical resource asymmetry. No significant differences are observed in handling varying file sizes or transmission durations. Packet loss and delays remain negligible due to the controlled testbed and small payload sizes.

In summary, while all evaluated algorithms are functional on resource-constrained platforms, CRYSTALS-Kyber offers the most efficient trade-off across execution time, power consumption, memory usage, and thermal behavior. BIKE and HQC remain viable alternatives under specific application constraints but present notable scalability challenges as security levels increase. These insights are critical for guiding PQC algorithm selection in future embedded and IoT deployments.

VII. Conclusion

In this study, we practically evaluate the performance of three post-quantum key encapsulation mechanisms – BIKE, HQC, and CRYSTALS-Kyber, on resource-constrained devices. Experimental results show that CRYSTALS-Kyber consistently achieves superior efficiency in execution time, power consumption, memory usage, and thermal behavior, making it the most suitable candidate for embedded and IoT platforms. While BIKE and HQC remain viable under specific conditions, their higher resource demands, particularly at increased security levels, present scalability challenges. These findings highlight the critical need to balance cryptographic strength with system-level constraints when selecting PQC algorithms for future deployments at scale.

Acknowledgments. Jesus Lopez is funded by the Department of Computer Science at UTEP. Viviana Cadena is funded by the Cyber Halo Innovation Research Program (CHIRP) at Pacific Northwest National Laboratory (PNNL).

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[2] P. Morgner and Z. Benenson, "Exploring security economics in iot standardization efforts," in *Proceedings 2018 Workshop on Decentralized IoT Security and Standards*, ser. DISS 2018. Internet Society, 2018.

[3] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "Iot: Internet of threats? a survey of practical security vulnerabilities in real iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182–8201, 2019.

[4] F. Thabit, O. Can, A. O. Aljahdali, G. H. Al-Gaphari, and H. A. Alkhzaimi, "Cryptography algorithms for enhancing iot security," *Internet of Things*, 2023.

[5] S. Subramani and S. K. Svn, "Review of security methods based on classical cryptography and quantum cryptography," *Cybernetics and Systems*, 2025.

[6] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, 1999.

[7] D. J. Bernstein and T. Lange, "Post-quantum cryptography," *Nature*, vol. 549, pp. 188–194, 2017.

[8] D. J. Bernstein *et al.*, "Introduction to post-quantum cryptography," *Post-quantum cryptography*, 2009.

[9] M. Mosca, "Cybersecurity in an era with quantum computers: Will we be ready?" *IEEE Security & Privacy*, 2018.

[10] National Institute of Standards and Technology, "Nist releases first 3 finalized post-quantum encryption standards," August 2024, accessed: 2025-03-27.

[11] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, Aug. 2018. [Online]. Available: https://www.rfc-editor.org/info/rfc8446

[12] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key {Exchange—A} new hope," in *USENIX Security Symposium (USENIX Security)*, 2016, pp. 327–343.

[13] N. Aragon, P. Barreto, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, S. Ghosh, S. Gueron, T. Güneysu *et al.*, "BIKE: bit flipping key encapsulation," 2022.

[14] C. A. Melchor, N. Aragon, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, E. Persichetti, G. Zémor, and I. Bourges, "Hamming quasi-cyclic (hqc)," *NIST PQC Round*, 2018.

[15] R. Avanzi, J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "Kyber: Module-lattice-based key encapsulation mechanism," CRYSTALS Project, Tech. Rep. Round 3, August 2021, nIST Post-Quantum Cryptography Standardization Project.

[16] H. Gharavi, J. Granjal, and E. Monteiro, "Post-quantum blockchain security for the internet of things: Survey and research directions," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 3, pp. 1748–1774, 2024.

[17] Z. Liu, K.-K. R. Choo, and J. Grossschadl, "Securing edge devices in the post-quantum internet of things using lattice-based cryptography," *IEEE Communications Magazine*, 2018.

[18] National Institute of Standards and Technology, "Post-quantum cryptography: Selected algorithms," 2024, accessed: 2025-03-25. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/selected-algorithms

[19] T. Monz, D. Nigg, E. A. Martinez, M. F. Brandl, P. Schindler, R. Rines, S. X. Wang, I. L. Chuang, and R. Blatt, "Realization of a scalable shor algorithm," *Science*, vol. 351, no. 6277, pp. 1068–1070, 2016.

[20] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.

[21] D. Sarah and C. Peter, "On the practical cost of grover for aes key recovery," in *Presentation at the 5th NIST PQC standardization conference*, 2024.

[22] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[23] T. M. Fernández-Caramés, "From pre-quantum to post-quantum iot security: A survey on quantum-resistant cryptosystems for the internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6457–6480, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8932459

[24] S. Ebrahimi, S. Bayat-Sarmadi, and H. Mosanaei-Boorani, "Post-quantum cryptoprocessors optimized for edge and resource-constrained devices in iot," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5500–5507, 2019.

[25] G. Tasopoulos, C. Dimopoulos, A. P. Fournaris, R. K. Zhao, A. Sakzad, and R. Steinfeld, "Energy consumption evaluation of post-quantum tls 1.3 for resource-constrained embedded devices," in *Proceedings of the 20th ACM International Conference on Computing Frontiers*, 2023, pp. 366–374.

[26] T. Liu, G. Ramachandran, and R. Jurdak, "Post-quantum cryptography for internet of things: A survey on performance and optimization," 2024.

[27] I. Fitzgibbon and C. Ottaviani, "Constrained device performance benchmarking with the implementation of post-quantum cryptography," *Cryptography*, vol. 8, no. 1, 2024. [Online]. Available: https://www.mdpi.com/2410-387X/8/1/21

[28] *A Performant Quantum-Resistant KEM for Constrained Hardware: Optimized HQC*. SCITEPRESS - Science and Technology Publications, Sep. 2024. [Online]. Available: https://enac.hal.science/hal-04699351

[29] D. Soni and R. Karri, "Efficient hardware implementation of pqc primitives and pqc algorithms using high-level synthesis," in *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2021, pp. 296–301.

[30] D. McGrew and J. Viega, "The galois/counter mode of operation (gcm)," National Institute of Standards and Technology, Tech. Rep., 2005, submission to NIST Modes of Operation Process. [Online]. Available: https://csrc.nist.gov/projects/block-cipher-techniques/bcm/modes/gcm/gcm-spec.pdf

[31] M. Dworkin, "Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac," National Institute of Standards and Technology, Gaithersburg, MD, NIST Special Publication 800-38D,

2007. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf

[32] P. Kampanakis, M. Campagna, E. Crocket, A. Petcher, and S. Gueron, "Practical challenges with AES-GCM and the need for a new cipher," in *Proceedings of the Third NIST Workshop on Block Cipher Modes of Operation*, March 2024.

[33] K. Bhargavan, C. Fournet, M. Kohlweiss, A. Pironti, and P.-Y. Strub, "Implementing tls with verified cryptographic security," in *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 445–459.

[34] S. Gueron, "Aes-gcm software performance on the current high end cpus as a performance baseline for caesar competition," *Directions in Authenticated Ciphers (DIAC)*, 2013.

[35] N. I. of Standards and Technology, "Secure hash standard (shs)," U.S. Department of Commerce, Gaithersburg, MD, Federal Information Processing Standards Publication FIPS PUB 180-4, 2015.

[36] National Institute of Standards and Technology, "Module-Lattice-Based Key-Encapsulation Mechanism Standard," U.S. Department of Commerce, Tech. Rep. FIPS NIST FIPS 203, August 2024. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.203

[37] National Institute of Standards and Technology (NIST), "NIST Selects HQC as Fifth Algorithm for Post-Quantum Encryption," https://www.nist.gov/news-events/news/2025/03/nist-selects-hqc-fifth-algorithm-post-quantum-encryption, 2025, accessed: 2025-07-10.

[38] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-kyber: a cca-secure module-lattice-based kem," in *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2018.

[39] Mbed TLS Development Team, "Mbed tls," https://github.com/Mbed-TLS/mbedtls, 2024, accessed: 2025-04-11.

[40] Open Quantum Safe Project, "liboqs: C library for quantum-resistant cryptographic algorithms," https://github.com/open-quantum-safe/liboqs, 2024, accessed: 2025-04-11.

[41] N. I. of Standards and Technology, "Post-Quantum Cryptography: Call for Proposals," https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf, December 2016, accessed: 2025-04-13.

[42] Wireshark Foundation, "Wireshark: Network protocol analyzer," https://www.wireshark.org/, 2024, accessed: 2025-04-11.

[43] Z. Ni, Z. Zhao, J. Jiang, D. Chen, D. Liu, S. Li, and W. Zhang, "HPKA: A High-Performance CRYSTALS-Kyber Accelerator Exploring Efficient Pipelining," in *2023 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, 2023, pp. 81–88.

[44] M.-S. Chen, T. Chou, and M. Krausz, "Optimizing bike for the intel haswell and arm cortex-m4," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2021, no. 3, pp. 97–124, 2021. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8969