IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 February 2025, accepted 14 March 2025, date of publication 24 March 2025, date of current version 6 May 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3554010

RESEARCH ARTICLE

Enabling Quantum-Resistant EDHOC: Design and Performance Evaluation

LIDIA POCERO FRAILE[1], GEORGIOS TASOPOULOS[2], CHRISTOS KOULAMAS[1], (Senior Member, IEEE), RAYMOND K. ZHAO[3], NAZATUL HAQUE SULTAN[3], FRANCESCO REGAZZONI[2,4], (Member, IEEE), AND APOSTOLOS P. FOURNARIS[1], (Member, IEEE)

[1]Industrial Systems Institute, R. C. Athena, 265 04 Patras, Greece
[2]Informatics Institute, University of Amsterdam, 1098 XH Amsterdam, The Netherlands
[3]CSIRO's Data61, Marsfield, NSW 2122, Australia
[4]Università della Svizzera italiana, 6900 Lugano, Switzerland

Corresponding author: Lidia Pocero Fraile (pocero@isi.gr)

This work was supported by the National Recovery and Resilience Plan Greece 2.0, funded by the European Union–NextGenerationEU, under Agreement TAEDR-0535864, project "Greece4.0." The work of Christos Koulamas and Apostolos P. Fournaris was supported by the European Union's Horizon Europe Research and Innovation Program Secure Open Source Software and Hardware Adaptable Framework (SecOPERA) under Grant 101070599.

ABSTRACT The Ephemeral Diffie-Hellman over COSE (EDHOC) is a compact and lightweight key establishment protocol for constrained scenarios that provides end-to-end application layer security context. However, because of the vulnerability of the discrete logarithm problem to Shor's algorithm, a quantum resistant replacement for Diffie-Hellman is required. This paper proposes an architecture to transform EDHOC into a quantum-resistant protocol, with an open-source implementation supporting various Post Quantum Cryptography (PQC) schemes from the National Institute of Standards and Technology (NIST) PQC standardization process. Necessary modifications to EDHOC are analyzed, integrating PQC Key Encapsulation Mechanisms (KEMs) and PQC digital signatures in a complete Post-Quantum (PQ) version of the EDHOC protocol (PQ-EDHOC). PQC operations often involve complex computations and require larger byte sizes, which can challenge resource-constrained devices designed for lightweight operation in constrained network environments. To evaluate the applicability of various PQC schemes in a realistic PQ EDHOC environment, various PQC KEM and Digital Signature combinations are assessed for execution time, energy consumption, memory usage, and network performance on an nRF52840 ARM Cortex-M4 platform with a 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) over BLE (Bluetooth LowEnergy) network. Results indicate that NIST standardized ML-KEM emerges as the only suitable KEM choice for resource-constrained environments, while promising new signature schemes, like HAWK, show significant performance improvement over the NIST standardized ML-DSA and selected for standardization FALCON.

INDEX TERMS Constrained embedded systems, EDHOC, post-quantum cyptography transition.

I. INTRODUCTION

Quantum computing represents a transformative leap in computational technology, surpassing the limits of classical systems by leveraging the principles of quantum mechanics to address specific complex problems at unprecedented speeds. This anticipated advancement, however, introduces significant risks to current cryptographic systems. Classical public-key cryptographic systems, such as Rivest-Shamir-Adleman Algorithm (RSA), Diffie-Hellman (DH), Elliptic Curve Diffie-Hellman (ECDH) and Elliptic Curve Digital Signature Algorithm (ECDSA), rely on the computational complexity of problems like integer factorization and discrete logarithms. However, these assumptions are undermined by Shor's algorithm [1], which runs on a quantum computer. Similarly, symmetric-key cryptographic systems, such as

The associate editor coordinating the review of this manuscript and approving it for publication was Lukasz Wisniewski.

AES and cryptographic hash functions, experience a reduction in security due to Grover's quantum search algorithm. Although doubling key sizes in symmetric-key cryptographic algorithms like AES is expected to maintain their security against quantum attacks, the overall resilience of classical cryptographic systems remains in question in the face of advancing quantum technology.

Post-quantum cryptography (PQC) has become a major focus for researchers aiming to protect digital systems from future quantum computer-based attacks. The U.S. National Institute of Standards and Technology (NIST) has been leading an open process to evaluate and standardize cryptographic algorithms that can resist quantum computers. Recently, NIST concluded the process and released three new Federal Information Processing Standards (FIPS): FIPS 203 [2], FIPS 204 [3], and FIPS 205 [4]. FIPS 203 focuses on a module-lattice-based key encapsulation mechanism (ML-KEM), an adaptation of the CRYSTALS-Kyber Key Encapsulation Mechanism (KEM). FIPS 204 introduces a module-lattice-based digital signature algorithm (ML-DSA), based on CRYSTALS-Dilithium. Meanwhile, FIPS 205 defines a stateless hash-based digital signature algorithm (SLH-DSA), derived from SPHINCS+. Additionally, NIST has launched an initiative to seek further post-quantum digital signature schemes, aiming to expand beyond lattice- and hash-based solutions [5]. This call focuses on alternatives that avoid structured lattice foundations, targeting designs with compact signatures and efficient verification speeds. As of October 2024, the selection process has advanced to the second round, with 14 candidates chosen based on several evaluation criteria [6].

Alongside NIST, the Internet Engineering Task Force (IETF) is also actively working to apply PQC to address the security challenges introduced by quantum computing in Internet security. Given the complexity of fully migrating existing systems to PQC, most works target a hybrid approach, integrating PQC algorithms with classical cryptographic algorithms in current protocols. This approach facilitates a smoother transition to PQC while ensuring that systems remain secure if vulnerabilities in the relatively new PQC algorithms are discovered in the future. The Post-Quantum Use in Protocols (PQUIC) IETF Working Group (WG) contributes to this transition by developing operational and design guidelines. Their ongoing drafts include *Terminology for Post-Quantum Traditional Hybrid Schemes* [7], which defines key terms for hybrid schemes; *Post-Quantum Cryptography for Engineers* [8], which examines the impact of quantum computing and challenges in transitioning to PQC; and *Hybrid Signature Spectrum* [9], which explores design goals and security considerations for hybrid digital signature schemes.

Although PQC provides security against quantum adversaries, it introduces several key challenges compared to classical cryptographic systems. One of the primary concerns is the increased computational, storage, and communication overhead [10]. Many PQC algorithms require significantly larger key sizes and more complex mathematical operations than traditional systems like RSA or ECC [11]. This can result in slower processing times and higher memory consumption, which is particularly problematic for resource-constrained devices such as IoT (Internet of Things) sensors, embedded devices, and similar low-power platforms [11]. Therefore, it is crucial to carefully assess the impact of these new cryptographic primitives on computation, storage, and communication overhead within existing protocols to ensure their practical adoption. In recent years, the PQC adaptation of TLS 1.3 (Transport Layer Security version 1.3) has undergone extensive benchmarking in various environments, showcasing how the integrated PQC algorithms affect TLS performance. Notable efforts in this area include works such as [12], [13], [14], [15], [16], [17], [18], and [19] and are analyzed in detail in Section II. However, as far as we know, no similar studies have been conducted for the Ephemeral Diffie-Hellman Over COSE (EDHOC) protocol.

EDHOC is a lightweight authenticated key exchange protocol developed by the Lightweight Authenticated Key Exchange (LAKE) WG of IETF, tailored to IoT devices with strict resource limitations [20]. It supports multiple authentication methods, such as signatures and static long-term ECDH keys, enabling versatile security solutions for IoT applications. However, the reliance of EDHOC on ECC exposes it to quantum attacks, as quantum computers can solve the EC discrete logarithm problem on which ECC is based. As quantum computing advances, adapting EDHOC to include PQC algorithms becomes essential to ensure its future resilience.

Integrating PQC into EDHOC offers the potential for enhanced security but also raises important performance considerations for IoT devices. Given the higher resource demands of PQC, evaluating its impact on computational efficiency, storage, energy, and communication is necessary to determine feasibility. Rigorous benchmarking enables us to explore the balance between improved security and performance limitations, which are critical for implementing practical quantum-resistant protocols in IoT. This paper presents our integration of PQC algorithms into the EDHOC protocol and shares detailed performance evaluations, assessing effects on computational load, storage, energy use, and communication. Our findings provide valuable insights for deploying quantum-resistant protocols in constrained environments, showing that a PQC-enhanced EDHOC can maintain security and efficiency. The major contributions of this paper are as follows:

- **Architectural Adaptation and Implementation of Quantum-resistant EDHOC**. A complete PQ-EDHOC protocol is implemented in the `uOSCORE-uEDHOC` [21] library, utilizing KEMs for key agreements and PQC digital signatures for authentication.
- **Adoption and Integration of Various PQC KEMs and Digital Signatures on EDHOC** for x86 systems using the `liboqs` [22] and `mupq` libraries, as well as

on ARM cortex-M4 microcontrollers using `pqm4` [23] and `PQClean` [24].

- **Evaluation and Performance Analysis of PQ-EDHOC for Constrained Embedded Systems**. A comprehensive comparison of performance impact is provided, in terms of execution speed, memory usage, and energy consumption. All the experiments are conducted on the nRF52840 DK (Cortex-M4) platform running Zephyr OS.
- **Evaluation of the Performance Impact of PQ-EDHOC in Real-world IoT Scenarios**: Specifically on constrained devices, such as the nRF52840 running Zephyr OS (Operating System), and constrained networks like 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) over Bluetooth Low Energy (BLE), analyzing execution speed, memory usage, and energy consumption during the PQ-EDHOC handshake.
- **PQ-EDHOC Performance Analysis under Different Network Conditions**. The handshake time of PQ-EDHOC with different PQC cipher suites is evaluated under various (emulated) network latencies and packet loss rates.

For a detailed explanation of technical terms, abbreviations and acronyms we refer the reader to a Glossary of Terms in the Appendix B and a Table with Abbreviations and Acronyms in Appendix C.

II. RELATED WORK

This section presents a brief overview of significant contributions in the literature related to benchmarking efforts in PQC.

A. BENCHMARKING OF NIST CANDIDATE PQC ALGORITHMS

In [25], Kannwischer et al. provided a detailed analysis of the implementation and performance evaluation of NIST PQC algorithms on the ARM Cortex-M4 microcontroller. The authors introduced the `pqm4` framework, a testing and benchmarking platform specifically tailored for evaluating the efficiency and feasibility of various PQC schemes in resource-constrained environments. The study meticulously examines the performance metrics of different algorithms, including their computational overhead, memory requirements, and execution time, thereby offering a comprehensive comparison of the candidates recommended by NIST. In [26], Kannwischer et al. presented a thorough evaluation of various post-quantum signature schemes recommended by NIST, specifically focusing on their performance on microcontroller platforms. Utilizing the `pqm4` framework, the authors benchmark multiple signature candidates based on their computational efficiency, memory usage, and overall feasibility for implementation in resource-constrained environments. In [27], Saarinen et al. provides an in-depth analysis of the energy consumption associated with implementing NIST's post-quantum cryptographic standards on mobile devices. The authors investigate the energy efficiency of various post-quantum algorithms, examining their performance in real-world mobile environments where power consumption is a critical concern. In [28], Vidakovic et al. evaluated NIST-selected PQC digital signatures in resource-constrained environments.

B. BENCHMARKING OF PROTOCOLS TRANSITIONING TO PQC

In [12], Paquin et al. presented a comprehensive evaluation of the performance implications of integrating PQC algorithms into the TLS 1.3 protocol. The study systematically benchmarks several NIST candidate PQC KEMs and digital signature schemes within a TLS framework, providing crucial insights into their computational overhead compared to classical algorithms. In [13], Sikeridis et al. conducted another in-depth analysis of the integration of PQC algorithms into the TLS 1.3 protocol, specifically focusing on the authentication phase. They evaluated various NIST candidate PQC signature schemes, assessing their performance regarding latency, computational overhead, and compatibility with existing TLS 1.3 implementations. In [14], Steinbach et al. successfully integrated the Kyber KEM and the SPHINCS+ digital signature scheme into the mbedTLS library, a widely used toolkit for securing communication in embedded devices and provided a comprehensive performance evaluation. In [15] and [17], Tasopoulos et al. presented in-depth analyses of the integration and performance implications of NIST PQC algorithms within the TLS 1.3 framework, specifically tailored for resource-constrained embedded systems. A similar effort was made in [29]. In [18], Gonzalez et al. presented a comprehensive analysis of the performance characteristics of KEMTLS compared to traditional post-quantum TLS implementations specifically tailored for embedded systems. In [19], Henrich et al. investigated the performance implications of integrating PQC KEMS into the TLS 1.3 protocol, specifically under diverse network conditions. The authors conducted a detailed evaluation of how different network characteristics—such as latency, bandwidth fluctuations, and packet loss—affect the performance of TLS 1.3 when enhanced with PQC KEMs. In [30], Samandari and Gritti integrated the post-quantum digital signature scheme CRYSTALS-Dilithium into Message Queue Telemetry Transport (MQTT) protocol to enhance authentication. They also provided a performance evaluation, focusing on CPU, memory, and disk usage. In [31], Manila et al. proposed another enhanced version of MQTT that integrates PQC techniques to ensure secure communication in IoT environments. In [32], Sikeridis et al. evaluated the protocol handshake performance when both post-quantum key exchange and authentication are integrated into TLS and Secure Shell (SSH) protocol. In [33], Kempf et al. conducted a detailed evaluation of how PQC affects the performance of the QUIC protocol.

C. IMPLEMENTATION AND BENCHMARKING OF EDHOC PROTOCOL

In [34], Hristozov et al. investigated the performance and resource implications of implementing two critical security protocols: Object Security for Constrained RESTful Environments (OSCORE) and EDHOC in resource-limited environments. It offers a detailed examination of the computational and memory costs associated with these protocols, highlighting the trade-offs between security and resource consumption that developers face when deploying them in constrained devices, such as those in the IoT ecosystem. The authors present empirical results demonstrating how OSCORE and EDHOC perform under various operational conditions, which aids in understanding their suitability for different applications. In [35], Pocero et al. presented a comprehensive study on developing and assessing an embedded module specifically designed to implement the EDHOC protocol. The authors detail the architectural design considerations that ensure compatibility with resource-constrained devices, such as those commonly used in the IoT ecosystem. By conducting rigorous performance evaluations, the paper highlights the module's efficiency in terms of computational overhead, memory usage, and communication latency. In [36], Hoglund et al. conducted a thorough review of the EDHOC protocol, offering extensive performance evaluation results that compare message sizes with those of Datagram TLS (DTLS). In [37], Fedrecheski et al. presented a comprehensive comparison between the EDHOC and DTLS protocols, focusing on message sizes and various authentication configurations to evaluate their computation time, energy consumption, and memory usage. They initially measured the handshake performance of both protocols on constrained hardware using an IEEE 802.15.4 radio and subsequently simulated their transmission times over LoRaWAN (Long Range Wide Area Network) networks. In [38], Perez et al. introduced "CompactEDHOC" as a lightweight alternative to the EDHOC protocol.

To the best of our knowledge, there has been no prior research focused on the implementation and benchmarking of the EDHOC protocol with the integration of NIST candidate PQC algorithms in resource-constrained devices. This work addresses this gap by incorporating PQC algorithms into EDHOC and evaluating its performance across various metrics within resource-constrained environments.

III. EPHEMERAL DIFFIE-HELLMAN OVER COSE (EDHOC)

EDHOC is a very compact and lightweight authenticated DH key exchange protocol for establishing a shared secret based on ephemeral keys. It provides identity protection, perfect forward secrecy, and mutual authentication based on out-of-band established credentials. The shared secret can be used to create an OSCORE [39] security context with shared keying material between the endpoints to provide application-layer protection, end-to-end encryption, integrity protection, and replay protection. The IETF LAKE WG describes the EDHOC protocol in [20], and it has recently been accepted as an RFC-proposed standard. The protocol is grounded in the Sign-then-MAC [40] variant of the SIGMA-I (Secure Internet Group Management Architecture - Version I) protocol and the Noise XX pattern within the Noise Protocol Framework [41], utilizing the ECDH algorithm. It follows the key establishment schemes outlined in NIST SP-800-56A [42], based on the discrete logarithm problem over elliptic curves. Furthermore, it employs HMAC-based Extract-and-Expand Key Derivation Function (HKDF) [43] to take the initial keying material and derive from it one or more cryptographically strong secret keys, and Concise Binary Object Representation (CBOR) [44] for data encoding.

EDHOC operates using a mandatory three-message exchange protocol (MSG1, MSG2, MSG3), with an optional fourth message (MSG4) for negotiating security parameters and establishing keys. The participants in this exchange are called the Initiator (I) and Responder (R). Each party generates a new ephemeral ECDH key pair for every session, shares the public components of their ECDH keys, computes a shared secret, and derives symmetric application keys from this secret.

The reference protocol supports four distinct authentication methods for establishing secure communication between two parties, utilizing combinations of signature and static DH keys. When employing a static DH key, authentication is achieved through a Message Authentication Code (MAC) derived from an ephemeral-static ECDH shared secret. The security properties of EDHOC depend on the methods used. Method 0, where digital signatures are used for both parties, exhibits security properties similar to those of the SIGMA-I protocol. Method 3, which uses MAC authentication for both parties, achieves security properties comparable to the Noise XX patterns [20].

The protocol also incorporates CBOR (Concise Binary Object Representation) Object Signing and Encryption (COSE) [45] for cryptographic operations, particularly for the identification of credentials such as including Internet X.509 Public Key Infrastructure Certificate, CBOR Encoded X.509 Certificates (C509) certificate, CBOR Web Token (CWT) and CWT Claims Set (CCS). Specific protocol message fields are designated for identifying and optionally transporting these credentials. If authentication credentials are pre-provisioned or acquired out-of-band over less constrained links, X.509 certificates can be identified by their hash value using the *x5t* COSE parameter. Alternatively, when both parties trust a shared authority, a chain of X.509 certificates can be included using the *x5chain* (*x5c*) COSE parameter.

EDHOC is not tied to a specific transport layer; however, the RFC recommends using a reliable transport mechanism, such as CoAP (Constrained Application Protocol) [46] in reliable mode. This default mechanism ensures packet ordering and addresses message duplication. In the forward message flow, that protects the client identity against active attackers and the server identity against passive attackers,

the CoAP client is the Initiator and the CoAP server is the Responder. EDHOC messages MSG1 and MSG3 are transmitted from the Initiator via `POST` requests to a well-known URI path, while the Responder sends MSG2 in an `ACK 2.04 (Changed)`. If message fragmentation is necessary, the EDHOC messages may be fragmented using the CoAP Block-Wise Transfer mechanism. [47].

IV. QUANTUM-RESISTANT EDHOC ARCHITECTURE

This section describes the architectural changes required to make the EDHOC protocol quantum-resistant. It also discusses the selected PQC algorithms enabling this transition and presents the rationale behind these choices. Finally, the introduction of unique EDHOC cipher suites incorporating PQC algorithms is detailed.

A. ARCHITECTURAL ADAPTATIONS

To fully support the EDHOC PQC transition, several necessary modifications to the protocol structure need to be made. The PQC KEM must be integrated as an alternative for the ECDH key agreement. On top of that, to provide a complete PQC-enhanced EDHOC protocol (PQ-EDHOC), authentication needs to be supported through PQC digital signatures to achieve full quantum-resistant security.

1) KEY DERIVATION

EDHOC is currently only specified for use with key exchange algorithms based on ECDH curves, but any KEM, including PQC KEMs, can be used. To adapt the protocol to the post-quantum environment the key exchange mechanism of EDHOC is transformed into a KEM scheme through an architectural adaptation, following the process described in the Post-Quantum Considerations section of the EDHOC RFC [20]. The message interaction must be modified by replacing the public DH key (`GX`) of the Initiator with the PQ ephemeral public key of the Initiator (`eph_pk`) and the public DH key of the Responder (`GY`) with the key encapsulation (`enc_eph`). Then both parties can compute a common shared secret (`ss_eph`) and use it on the protocol instead of the `GXY` shared secret computed by DH.

2) AUTHENTICATION

EDHOC authentication method 0 employs signature-based authentication for both parties and relies on COSE for credential identification, supporting a wide range of signature algorithms and credential types. We selected this specific method as the first approach to quantum transit the protocol, aligning with the specifications outlined in [48] and maintaining compatibility with existing implementations by adding new cipher suites, as described later in Section IV-C. In this work, the EDHOC protocol is adapted to work with PQ-X.509 certificates according to RFC 9360 [49], which addresses the COSE structure for X.509 certificates. Two scenarios are considered: i) where the certificates are identified by a hash value using the *x5t* COSE parameter, and ii) where both parties exchange a unique PQ-X.509 certificate using the *x5chain* COSE parameter.

The overall quantum-resistant adapted handshake message exchanges are presented in detail in Fig. 1, highlighting the PQ operations performed in each phase. Presented in blue are the PQ KEM-related operations, and in green are the PQ Digital signature ones. Initially, the Initiator generates a PQ key pair and sends the ephemeral public key (`pk_eph`) to the responder as part of MSG1. The Responder, using the Initiator's ephemeral public key, calls the PQ Encapsulation function that produces a Ciphertext (`ct_eph`), which is sent to the Initiator as part of the MSG2, and a Shared Secret (`ss_eph`), which it keeps since this is the actual shared secret. The Initiator upon receiving the Ciphertext calls the Decapsulation function with its Secret Key and produces the same Shared Secret as the Responder. At this point, both the Initiator and the Responder share the same Shared Secret, which is used to derive a fixed-length, uniformly pseudo-random key (PRK) resulting from a completed EDHOC session. The Initiator can now decrypt the remainder of MSG2, a ciphertext encrypted with the PRK shared key. It extracts and validates every element of the received MSG2, and retrieves the authentication credentials of the Responder from the `ID_CRED_R`. If it contains a PQ-X509 certificate, the initiator verifies it using the pre-provisioned certificate authority's (CA) public key. Once the Initiator gets the Responder's authentication key, it can verify the PQ signature included on MSG2. Then the Initiator proceeds with the generation of MSG3 by computing its PQ signature and encrypting the corresponding message fragment with the AEAD protocol and the shared PRK derived from the PQ KEM. Finally, the Responder receives MSG3, decrypts the message with AEAD (Authenticated Encryption with Associated Data) protocol and the corresponding PRK, recovers the Initiator's authentication credentials, and authenticates it.

B. SELECTION OF PQC KEMs AND DIGITAL SIGNATURE ALGORITHMS

To transform EDHOC to be quantum-resistant, several replacement algorithms have been selected for the EDHOC key exchange (that is now transformed into a KEM) and digital signatures. Following this selection, the PQC performance overhead in EDHOC must be evaluated in real-world IoT scenarios where constrained devices communicate over constrained networks. The primary consideration in these scenarios is the compactness of the produced artifacts of the PQC scheme, meaning keys, signatures, and ciphertexts. The memory consumption of the implementation is also considered, as excessive usage may prevent the scheme from being practically evaluated in real-world scenarios. The algorithms are selected from the NIST PQC process and from the recently standardized algorithms. Some promising, constrained environment-friendly algorithms are also included, that advanced to the still-open 4th NIST PQC round for further evaluation. Beyond that, the paper

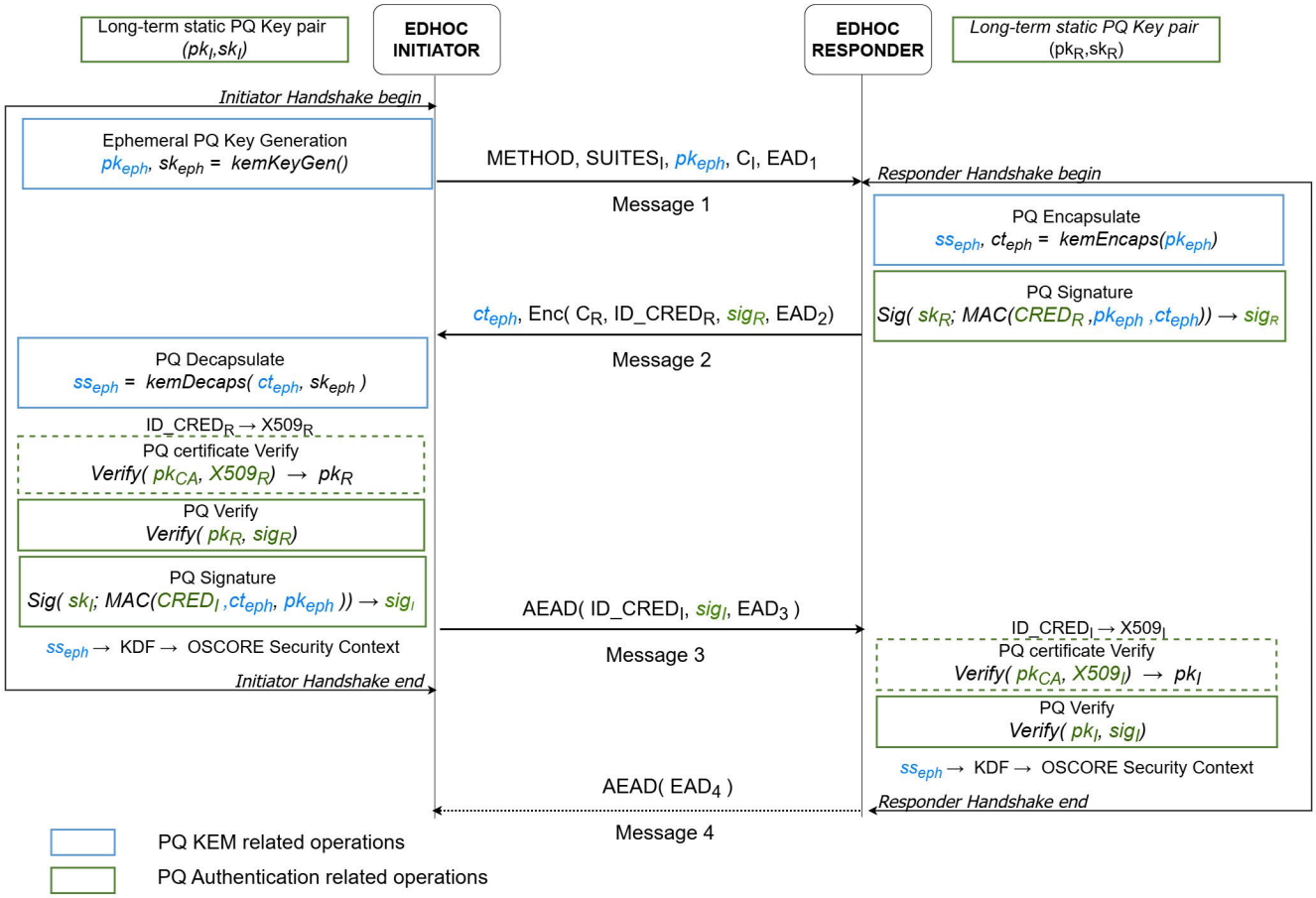


FIGURE 1. PQ-EDHOC handshake messages.

extends its study to include signature schemes submitted to the latest NIST call for additional PQC digital signatures, which feature no schemes based on structured lattices, short signatures, and efficient verification time (currently in the 2nd round). Additionally, the results from [26], which evaluate the suitability and performance of round 1 candidates for Arm Cortex-M4 microcontrollers, have been considered in selecting additional digital signature algorithms for PQ-EDHOC.

NIST introduced five security levels for the algorithms in the PQC standardization process. Security levels 1,3, and 5 correspond to resistance against attacks on that PQC algorithm (with a specific parameter set) that would require the same or more resources as a key search on AES-128, AES-192, and AES-256, respectively. Levels 2 and 4 correspond to resistance against attacks requiring the same or more resources as a collision search on SHA-256 and SHA-384, respectively. Finally, algorithms with o quantum attack resistance are categorized as security level 0. In this paper low-security level versions of the PQC algorithms are considered, as they offer smaller artifact sizes and provide sufficient security for EDHOC, primarily used in embedded IoT systems.

1) INCLUDED KEMs

ML-KEM (based on CRYSTALS-Kyber) [2] that is the NIST standardized PQ KEM, is included in our implementation, with parameters for NIST security level 1 (ML-KEM-512) and 3 (ML-KEM-768). To cover a broader range of PQC KEMs, NIST PQC round 4 candidates were also considered. Specifically, HQC and BIKE [50] were included, with parameters for NIST security level 1 (HQC1 and BIKE1). Classic McEliece [51] was not evaluated because its large keys could not fit in our experimental platforms. SIKE [52] was not considered, since the underlying security assumptions have been completely broken [53], [54].

2) INCLUDED DIGITAL SIGNATURES

ML-DSA (based on CRYSTALS-Dilithium) [3] that is the NIST standardized digital signature, is included in our implementation, with parameters for the lowest NIST security level 2 (ML-DSA-44). Additionally, FALCON [55] with parameters for NIST security level 1 (FALCON1) is included. FALCON has been selected for standardization by NIST, but an initial public draft has yet to be released. Finally, SLH-DSA is not considered, as the signature size (7856 in NIST security level 1) is prohibitive for constrained networks.

To diversify the PQC digital signatures evaluated in the EDHOC protocol, the NIST PQC additional round for digital signatures candidates was considered [6]. Compact algorithms suited for constrained IoT scenarios were mainly evaluated. Table 1 compiles information from the corresponding submission documents and [56] to assess the compactness of these signature schemes. The table compares the lowest NIST security levels provided by each PQC algorithm, which are achieved using the displayed Parameter Set. To offer a quantitative way to measure and compare the compactness of these algorithms, the size of 4 digital signatures and 2 public keys is calculated in the last column, with algorithms ordered accordingly. This metric reflects a generic IoT authentication scenario where X.509 certificates are used as credentials between two entities that do not initially know each other but share trust in the same CA as a trust anchor, without intermediate trust chains. It should be noted that verifying the integrity and validity of certificates is out of the scope of EDHOC, and the choice of revocation mechanism is left to the Initiator and Responder application [20]. For specific scenarios where the authentication credentials of the other party can be acquired out-of-band, X.509 certificates may not need to be transmitted. In such cases, only the cost of the signature itself should be considered. We focus on the more general scenario, which is why algorithms with a $4sig+2pub$ size smaller than the standardized ML-DSA were selected. Note that schemes marked with an asterisk (*) did not advance to NIST Round 2.

Even though SQIsign is the most compact PQC algorithm, the lack of a 32-bit system implementation prohibited its usage in the proposed implementation. SNOVA's and MAYO's implementation required more memory than the paper's experimental platform capacity. EHTv3/EHTv4 has been susceptible to attacks on the underlying hard problem and thus not considered. Finally, HAWK [57] and HAETAE [58] were selected for evaluation due to their compactness compared to ML-DSA and because their implementation can run in our resource-constrained device. HAWK is used with NIST security level 1 parameters (HAWK1), while HAETAE is used with security level 2 (HAETAE2). Although HAETAE did not progress to NIST round 2 for additional signatures, it is recently selected for standardization by the Korean PQC process [59] and was therefore included in our evaluation.

C. EDHOC CIPHER SUITES

An EDHOC cipher suite consists of an ordered set of algorithms from the Internet Assigned Numbers Authority (IANA) COSE registry [60] as well as the EDHOC MAC length, and is identified with a predefined integer label in the IANA EDHOC registry [61]. Since PQC KEMs and signature algorithms under evaluation have not yet been registered in the IANA COSE registry, each new PQC algorithm must be assigned an available registry value. Furthermore, for each combination of PQC KEM and digital signature to be tested, a new cipher suite must be defined in the IANA EDHOC registry.

More specifically, an EDHOC cipher suite consists of: EDHOC AEAD algorithm, EDHOC hash algorithm, EDHOC MAC length in bytes, EDHOC key exchange algorithm, EDHOC signature algorithm, application AEAD algorithm, and application hash algorithm. All tested cipher suites, including the selected IANA COSE values for each PQ KEM algorithm and PQ digital signature in their corresponding sections, are presented in Table 2. All added cipher suites will maintain the same algorithms for the other parameters, specifically employing AES-CCM with a 16-62-128 configuration, SHA-256 for the hash, and an 8-byte MAC length.

V. IMPLEMENTATION

The open-source implementation of our complete proposal for PQ-EDHOC is available on GitHub,[1] including the full source code, documentation, and test benchmarks that we analyze in this paper. To develop it, the aforementioned architectural adjustments are implemented in the `uOSCORE-uEDHOC` library [21]. The `uOSCORE-uEDHOC` library is a lightweight C implementation of the IETF protocols EDHOC [20] and OSCORE [39] specifically designed for microcontrollers. This implementation is independent of the operating system, cryptographic engine, and the underlying transport protocol in the case of `uEDHOC`. Notably, it operates using only stack memory avoiding heap allocation. This C library has been successfully evaluated on a broad range of microcontrollers [34] and provides a straightforward way to incorporate new cryptographic engines by using a cryptographic software wrapper. This makes it the ideal foundation for our PQ transition implementation.

We integrated the PQC modules by using the cryptographic wrapper provided by `uOSCORE-uEDHOC`. This involved adding KEM functionalities, including PQ key generation, encapsulation, and decapsulation, along with PQ digital signature functionalities such as signature generation, verification, and PQ-x509 certificate verification. aLL necessary modifications to support KEMs and post-quantum digital signatures and certificates were also made, as described in Section IV-A. Furthermore, the library code was adjusted to ensure that buffer sizes can accommodate the larger data requirements of PQC compared to traditional Public Key Cryptography (PKC) while allowing buffer sizes to be defined at compilation time based on the selected PQC algorithms. This approach enables the selection of PQ-EDHOC cipher suites at compilation time, ensuring they match our evaluation platform's memory constraints.

Current PQC algorithms require handling large data sizes, increasing significantly the EDHOC message size. To evaluate in real-world constrained network scenarios, CoAP messages must be fragmented using the CoAP Block-Wise Transfer mechanism [47]. To achieve this, the `libcoap` [62]

[1] https://github.com/LPFraile/PQ-EDHOC

TABLE 1. Signature schemes from NIST's additional PQC round.

Scheme	Parameter Set	NIST Security Level	Public Key Size (Bytes)	Signature Size (Bytes)	$4sig + 2pub$ (Bytes)
ECDSA	NIST P-256	0	32	64	320
SQIsign	I	1	64	177	836
RSA	2048	0	272	256	1568
SNOVA	(24, 5, 16, 4)	1	1016	248	3024
MAYO	one	1	1168	321	3620
EHTv3/EHTv4 *	v4-1	1	1110	369	3696
HAWK	512	1	1024	555	4268
FALCON	512	1	897	666	4458
HAETAE *	120	2	992	1463	7836
ML-DSA	44	2	1312	2,420	12304

TABLE 2. Definitions of IANA EDHOC cipher suites for PQC.

PQC Combination PQ KEM/PQ Signature	Cipher Suite (IANA-COSE)	Cipher Suite (IANA EDHOC)
ML-KEM-512/FALCON1	-48 , -59	8
ML-KEM-768/FALCON1	-49 , -59	9
HQC1/FALCON1	-51 , -59	10
BIKE1/FALCON1	-55 , -59	11
ML-KEM-512/ML-DSA-44	-48 , -61	12
ML-KEM-512/HAWK1	-48 , -64	14
ML-KEM-512/HAETAE2	-48 , -65	15

library was integrated into the Linux-based executable, and the `coap-client` API was utilized from the Zephyr OS network subsystem for the embedded system. This included implementing the `Tx/Rx` EDHOC callback functions to handle the transmission and reception of messages.

By viewing it as a standalone module (e.g. a self-contained EDHOC single Initiator or Responder), the implementation of the PQ-EDHOC requires the supported IANA cipher suites and the PQ authentication credentials as mandatory inputs. To authenticate the other party in a mutual authentication process, the PQ-EDHOC module needs the authentication credentials of the other party to be pre-provisioned or acquired out-of-band, or the CA public key and CA certificate when public key infrastructure is used. The module derives a shared OSCORE security context for each party, consisting of an OSCORE shared secret and salt (referred to as the master secret and master salt in RFC 8613), using the EDHOCExporter interface. Additionally, it returns fixed-length, uniformly pseudo-random keys (PRK) derived from the PQ shared secrets, which can be used to generate a security context for other application security protocols, if necessary.

A. PQC CRYPTOGRAPHY ENGINES

To enable the EDHOC protocol to work with PQC algorithms, four different cryptographic engines were integrated into the PQ-EDHOC implementation to ensure compatibility across various platforms. The `liboqs` [22] and `mupq` [63] cryptographic engines are intended for use when the protocols run as a Responder on non-constrained servers, such as x86_64 architectures as well as for the network simulation benchmarking. On the other hand, `pqm4` [23] and `PQClean` [24] are employed in the Zephyr OS application running on Cortex-M4 embedded systems.

1) HIGH END SYSTEMS

PQ-EDHOC compatibility for both x86 systems and 64-bit ARM architectures, which can be built on Linux, macOS, and Windows, was achieved by integrating the `liboqs` [22] implementation into `uOSCORE-uEDHOC`. This open-source, multi-platform C library provides quantum-resistant cryptographic algorithms. It offers a collection of open-source implementations of quantum-resistant KEMs and digital signature algorithms through a common API, making it easy to switch between algorithms. The implementations are derived from the reference and optimized code submitted by various teams to the NIST PQC Standardization Project, sourced directly from these teams or via the `PQClean` project [24]. Specifically, the implementations of ML-KEM-512, ML-KEM-768, BIKE1, and HQC1 for PQC KEMs, and ML-DSA-44 and Falcon-padded-512 for digital signatures were integrated. Additionally, the plain C reference implementation for HAWK1 and HAETAE2, sourced from the NIST PQC process submission and found in the `mupq` repository [63], were included in our implementation.

2) CONSTRAINED EMBEDDED SYSTEMS

To optimize the protocol for constrained embedded systems based on the ARM Cortex-M4 platform, the open-source library `pqm4` [23] was integrated into our implementation as the main PQ cryptographic engine. The library is customized for high performance, memory efficiency, and seamless integration with embedded systems, making it a valuable resource for evaluating the feasibility of post-quantum cryptography on IoT and embedded devices. Designed to support research and benchmarking on resource-constrained devices, `pqm4` includes implementations of post-quantum KEMs and digital signature schemes, targeting the 32-bit ARM Cortex-M4 family of microcontrollers. It provides optimized implementations of symmetric cryptography primitives such as Keccak primitives, SHA-3 and SHAKE-256 [64] that are required in almost every PQC algorithm under this study. The library includes the latest versions of all the selected algorithms for standardization by NIST, part of the 4th round of the NIST PQC standardization process, and

also reference implementations for 15 submissions and M4-optimized implementations for five submissions in the first round of additional signatures of the NIST PQC standardization process [26]. Additionally, some PQC algorithms provide multiple implementations optimized toward different performance characteristics. We chose to integrate implementations that contain optimized assembly for the Cortex-M4, focusing on the `m4stack` versions to optimize memory consumption where applicable. Particularly, the `m4stack` implementation of ML-KEM-512, ML-KEM-768, and ML-DSA-44 is included in the PQ-EDHOC implementation. For BIKE1 and HAETAE2, the available implementation with Cortex-M4F-specific optimizations that typically leverage floating-point registers was utilized. However, `pqm4` does not provide optimized implementations for HQC1 and HAWK1 algorithms. Therefore, the reference implementation from `PQClean` for HQC1 and from `mupq` for HAWK1 was used in our work.

For the standardized PQC algorithms, ML-KEM and ML-DSA, an earlier version of those implementations was used corresponding to the initial public draft for each FIPS standard. Specifically, implementations from `pqm4`'s commit hash `8d44b72` and `liboqs`'s commit hash `d0353500` that implements the FIPS 203 [2] and FIPS 204 [65] initial public draft was integrated into the PQ-EDHOC implementation. Even though the final standards are now published and we are evaluating a previous version of our work, the performance differences are almost minimal. This can be validated by comparing `pqm4`'s benchmarks for the initial public draft version [66] with the final standard version [67] of FIPS 203 and 204.

VI. ANALYSIS AND EVALUATION OF PQ-EDHOC FOR CONSTRAINED DEVICES

For evaluating PQ-EDHOC, we selected an nRF52840 Development Kit (DK) board by Nordic Semiconductors, which features an nRF52840 microcontroller. This board is designed for ultra-low-power applications and provides hardware support for low-energy protocols, like BLE. It is equipped with an ARM Cortex-M4 running at 64 MHz, with 1 MB of Flash and 256 kB of RAM. The board also includes pins dedicated to power consumption measurements. A Cortex-M4-based board was chosen because it is the selected platform by NIST for evaluating PQC algorithms in resource-constrained scenarios [68], and it is widely used for embedded systems evaluation in the academic world. Additionally, the board's wireless multi-protocol support enables testing in a real-world IoT environment.

We run the PQ-EDHOC implementation over a Zephyr Real-time Operating System (RTOS). Zephyr RTOS provides a flexible interface for managing threads in constrained devices that may have real-time requirements. Using an RTOS enables a system to operate a network stack on top of the application's functionality and enables us to control the hardware for protocols like BLE.

All the software stack is written in C code and is built using GCC compiler version `11.4.0-1ubuntu1` with optimizations for size turned on (`-Os`). After cross-compiling, we flash the produced binary to the board for evaluation, using the J-TAG serial interface provided by the board.

The measurement process used for benchmarking speed, power, energy, overall code size, and memory usage is explained in detail in the A. Three experiments were conducted, and their results are presented in the following subsections. The first experiment evaluates the performance of the integrated PQC algorithm on our test board running Zephyr OS (Section VI-A). The second experiment measures the PQC overhead on constrained devices by running both parties of the implemented protocol on the same test board, with data exchanged between them via buffers(Section VI-B). The third experiment assesses the implementation in a real-world scenario by executing the protocol over a BLE-constrained network (Section VI-C).

A. PQC ALGORITHM PRIMITIVES EVALUATION

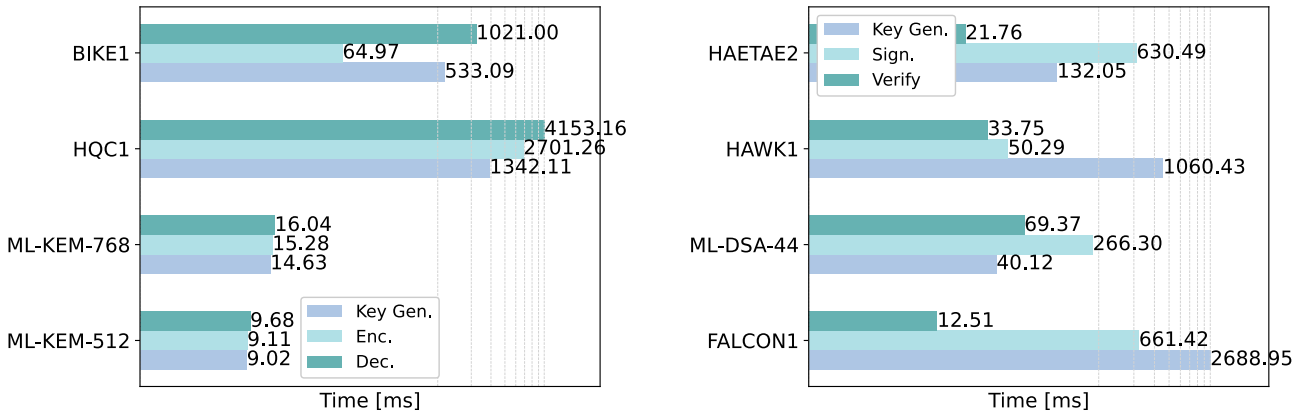
In this section, the evaluation results of the individual PQC operations for each algorithm under test on our nRF52840 DK board running Zephyr OS [69] are provided in detail. Tests have been conducted for the KEM primitives, which consist of key generation, encapsulation, and decapsulation, with each functionality measured independently. The same procedure was adopted to test signature schemes, including key generation, signing, and verification. The speed benchmarking results provide the average, minimum, and maximum clock cycle values for each operation in every algorithm, as shown in Table 3, along with the transmission byte sizes for the nRF52840 DK board when the core is running at 64 MHz. Fig. 2a shows the results for PQC KEM algorithms, while Fig. 2b shows results for PQC signature algorithms. Additional implementation metrics, such as code size and stack usage for PQC algorithm primitives, are beyond the scope of this work. However, detailed evaluations for these metrics on the Cortex-M4 platform have been reported in [25] and [26], and energy evaluations are available in [27] and [29].

As expected, the measurements presented here align with the prior works that have extensively evaluated these algorithms. It is worth noting that the execution time of some algorithms has a large variance. The execution time of Dilithium's Sign operation varies, due to the rejection sampling process [70]. Similar findings can be observed in the Key Generation operations of FALCON and HAWK, and HAETAE's Key Generation and Sign operations. Note that the Key Generation operation for digital signatures is never performed online, so it does not influence the execution time of any protocol where the primitive is being used.

Regarding the NIST round 4 candidates, HQC and BIKE, it can be observed that the sizes and the execution times are significantly larger than ML-KEM, even larger than the

TABLE 3. Average execution speed for PQC on the NRF52840 DK board.

Scheme	Transmission Bytes		Average Execution Speed (CLK)		
KEMs	**pk**	**ct**	**Key Gen.**	**Enc.**	**Dec.**
ML-KEM-512	800	768	AVG: 577397 MIN:575916 MAX:594376	AVG:583546 MIN:579552 MAX:603040	AVG:619874 MIN:618436 MAX:636816
ML-KEM-768	1184	1088	AVG:936640 MIN:933736 MAX:952528	AVG:978477 MIN:972976 MAX:996784	AVG:1026869 MIN:1023928 MAX:1042764
HQC1	2249	4433	AVG:85895120 MIN:85894964 MAX:85896524	AVG:172881156 MIN:172881156 MAX:172881160	AVG:265802284 MIN:265802284 MAX:265802284
BIKE1	1541	1573	AVG: 34118285 MIN:34118285 MAX:34118285	AVG:34118285 MIN:34118285 MAX:34118285	AVG:34118285 MIN:34118285 MAX:34118285
Digital Sign.	**pk**	**sign**	**Key Gen.**	**Sign.**	**Verify**
FALCON1	897	690	AVG: 172093414 MIN:791360 MAX: 210597568	AVG: 42330961 MIN:42134324 MAX: 42431628	AVG:801200 MIN: 791360 MAX:812228
ML-DSA-44	1312	2420	AVG:2568207 MIN:2549928 MAX:2705000	AVG:17043689 MIN: 5382760 MAX:60048672	AVG: 4439786 MIN: 4398932 MAX:4471428
HAWK1	1024	555	AVG:67867999 MIN: 2159304 MAX: 161382872	AVG:3218858 MIN: 3218732 MAX: 3218936	AVG: 2160464 MIN: 2159304 MAX: 2161380
HAETAE2	992	1474	AVG:8451183 MIN:1392636 MAX: 41111252	AVG: 40351579 MIN: 6887972 MAX: 219132624	AVG:1392892 MIN: 1392636 MAX: 1393296



(a) KEMs (b) Digital Signatures

FIGURE 2. Average execution speed of PQC algorithms.

ML-KEM-768 at NIST security level 3. We can roughly extrapolate the results to the execution time of a cipher suite in a real-world IoT scenario. The measurements in Section VI-B and Section VI-C validate our extrapolation.

B. PQ-EDHOC PERFORMANCE EVALUATION

This section evaluated the PQC performance overhead of PQ-EDHOC on embedded systems. Comparisons are made between the PQ-EDHOC performance with different PQC algorithms and the classic EDHOC using traditional public key cryptography algorithms. More specifically, comparisons are focused on the EDHOC cipher suite 2, referenced as mandatory implementation in the EDHOC RFC. This specific cipher suite uses ECDH over the NIST P-256 curve (secp256r1) for the key exchange algorithm and ECDSA over NIST P-256 with SHA-256 for the signature algorithm. Cipher suite 2 is referenced as secp256r1/ECDSA for the rest of the paper. All tests keep the same algorithms for the remaining elements of the cipher suite, AES CCM with 16-62-128 configuration, SHA-256 for hash, and 8-byte MAC length. Furthermore, the two authentication scenarios mentioned in Section IV-A2 are compared. The credential type *x5t* refers to the case where only the identification value of the certificate is sent, and the credential type *x5chain* indicates that the entire X.509 certificate is exchanged between the two parties.

To isolate the benchmark environment from other communication and network stack functionalities, the evaluations of RAM, FLASH, total handshake time, processing time, power, and energy consumption in this section were conducted offline in a unit test environment where no messages were transmitted over the network. For these tests, both Initiator and Responder were run on two separate threads that communicated via a message buffer, following the same procedure used in [34] to benchmark the basic uEDHOC implementation. To present results comparable to those in [34] for the cryptographic operations, the `TinyCrypt` [71] and `C25519` [72] libraries were utilized, which were specifically developed for microcontrollers.

1) SPEED BENCHMARKING

A speed benchmarking framework was used to evaluate the complete handshake process for Initiator and Responder separately, as illustrated in Fig. 1. For the Initiator, the handshake spans from ephemeral key generation to the final OSCORE security context. For the Responder, the complete handshake starts from the reception of message 1 until the derivation of the OSCORE security context. This evaluation focuses exclusively on the mandatory handshake messages, with optional message 4 intentionally omitted. Key confirmation is typically provided by sending an application message from the Responder to the Initiator, protected with a key derived from the Exporter, such as OSCORE. The results, shown in Table 4, include the average, maximum, and minimum clock cycle counts *CLK* for 100 interactions for most schemes, except for the HQC1/FALCON1 cipher suite, which is repeated only 10 times due to its slow execution time.

Firstly, it is observed that the total handshake time for the Initiator is consistently lower than that for the Responder, except for cipher suits with HQC1 and BIKE1, as is shown in Fig. 3 (left). This difference relates to the operations involved in each role's handshake process. Specifically, the Initiator generates an ephemeral key, waits for the Responder to encapsulate and sign MSG2, then decapsulates and verifies the Responder's signature before signing MSG3, without including the time that the Responder requires to verify MSG3. Conversely, the Responder's handshake time includes nearly all operations from both the Initiator and Responder, except for the ephemeral KEM key generation performed by the Initiator.

As can be observed in Fig. 3, for cipher suites with HQC and BIKE KEMs, the total time and energy consumption of the Initiator is much larger than that of the Responder, because the key generation is only performed in the Initiator side and it is much slower than in cipher suites with ML-KEM.

It can be observed from Table 4, that the HQC1/FALCON1 cipher suite relying on HQC1 KEM has a significantly long handshake time that may be unacceptable for various embedded system applications. There is currently no publicly available ARM Cortex-M4 optimized HQC1 implementation to consider for further exploring this issue. On the other hand, HAWK1 (e.g. in ML-KEM-512/HAWK1 cipher suite), although it relies on a non-optimized implementation has the fastest handshake time among all evaluated cipher suites in Table 4. As research progresses and better-optimized implementations are published, it can be expected that HAWK will perform even better.

TABLE 4. PQ-EDHOC total handshake time.

Cipher Suite	Cred. Type	Role	App MSG Size (Bytes)	Time (CLK) *AVG*	*MIN*	*MAX*
ML-KEM-512/FALCON1	*x5t*	In	1492	91329603	90850616	91716808
		Re	1447	93224779	92756820	93622728
	x5chain	In	3187	99424431	99017656	99692952
		Re	3146	104829815	104392320	105107048
ML-KEM-768/FALCON1	*x5t*	In	1872	93257949	92808744	93572732
		Re	1766	94669531	94213712	94995728
	x5chain	In	3570	101500743	101167960	101794584
		Re	3463	106546824	106183896	106849156
ML-KEM-512/ML-DSA-44	*x5t*	In	3254	53550150	30180948	136097388
		Re	3209	62446478	39070804	144989988
	x5chain	In	7149	73092318	51666008	143386776
		Re	7103	93045130	71623248	163353416
HQC1/FALCON1	*x5t*	In	2942	615839754	615758132	615967352
		Re	5111	532200253	532114700	532318368
BIKE1/FALCON1	*x5t*	In	2233	192256162	191919788	192553448
		Re	2253	160631836	160284668	160940168
ML-KEM-512/HAWK1	*x5t*	In	1389	13325356	13318224	13431492
		Re	1344	15987788	15981864	16059508
ML-KEM-512/HAETAE2	*x5t*	In	2308	84159395	28431080	173408732
		Re	2263	87365164	31637252	176615432
secp256r1/ECDSA	*x5t*	In	127	90084115	90084112	90084468
		Re	115	109907480	109907480	109907572
	x5chain	In	410	110941339	110767008	110943100
		Re	398	150694478	150691920	150947784

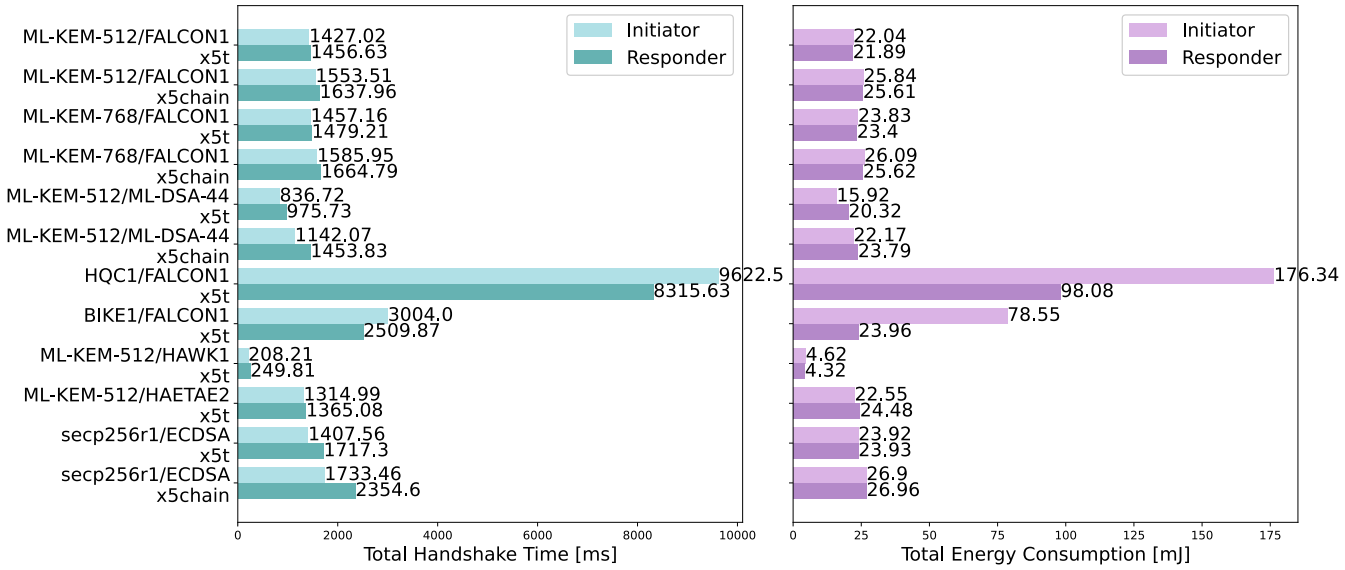


FIGURE 3. PQ-EDHOC total handshake time and energy consumption.

2) POWER AND ENERGY BENCHMARKING

The results from the power and energy benchmarking setting are reported in Table 5, where two oscilloscope triggers were employed to measure the performance of the Initiator and Responder threads separately. The table presents the average processing time in milliseconds (*ms*), the average power in milliwatts (*mW*), and the total energy consumed in millijoules (*mJ*) for both the Initiator and Responder. The power and energy consumption measurements correspond only to the processing time *without* measuring the consumption while waiting for the other party to execute the protocol and the overhead of exchanging messages.

When the protocol uses *x5t* credentials, it is observed that all PQC cipher suites consume less energy than the traditional cipher suite with ECDH and ECDSA. Specifically, ML-KEM-512/HAWK1 reduces energy consumption by approximately 81.9% for the responder and 80.6% for the initiator. ML-KEM-512/ML-DSA-44 consumes 15.1% and 33.4% less energy for Responder and Initiator, respectively. ML-KEM-512/FALCON1 consumes 8.5% and 7.8% less energy for Responder and Initiator, respectively. ML-KEM-768/FALCON1 and ML-KEM-768/HAETAE2 consume almost the same energy for the Responder and Initiator.

When the protocol incorporates *x5chain* credentials and transmits the entire X.509 certificate, it is observed that ML-KEM-768/FALCON1 consumes slightly less energy than the traditional cipher suite with ECDH and ECDSA. ML-KEM-512/ML-DSA-44 consumes approximately 11.7% and 17.6% less energy for the Responder and the Initiator, respectively.

3) MEMORY USAGE BENCHMARKING

Stack usage was measured by initializing the stack of the running thread with a specific pattern (`0xaa`) at boot time and then examining how many pattern bytes were overwritten after executing the protocol. This process allowed us to determine the amount of stack used for each test and we reported for both the Initiator and Responder in Table 7. These results allow us to define the stack size of Initiator and Responder threads to optimize RAM consumption for each

TABLE 5. PQ-EDHOC total average processing time, power and energy evaluation.

Cipher Suite	Cred. Type	Avg. Processing Time (ms)		Power (mW)		Energy (mJ)	
		In	*Re*	*In*	*Re*	*In*	*Re*
ML-KEM-512/FALCON1	*x5t*	772.93	767.68	28.51	28.51	22.03	21.89
	x5chain	892.42	887.07	28.96	28.87	25.84	25.61
ML-KEM-768/FALCON1	*x5t*	812.01	800.77	29.35	29.22	23.83	23.40
	x5chain	899.06	886.89	29.02	28.89	26.09	25.62
ML-KEM-512/ML-DSA-44	*x5t*	490.90	623.55	32.43	32.58	15.92	20.32
	x5chain	759.93	816.09	29.18	29.15	22.17	23.79
HQC1/FALCON1	*x5t*	6282.13	3485.33	28.07	28.14	176.34	98.08
BIKE1/FALCON1	*x5t*	2520.83	833.4	31.16	28.75	78.55	23.96
ML-KEM-512/HAWK1	*x5t*	141.85	135.92	32.57	31.78	4.62	4.32
ML-KEM-512/HAETAE2	*x5t*	702.04	771.42	32.12	31.74	22.55	24.48
secp256r1/ECDSA	*x5t*	860.65	871.59	27.79	27.45	23.92	23.93
	x5chain	991.12	1007.44	27.14	26.76	26.90	26.96

TABLE 6. Memory usage.

Cipher Suite	Cred. Type	Role	Total (Bytes)			PQ Crypto (Bytes)			PQEDHOC (Bytes)		
			text	*data*	*bss*	*text*	*data*	*bss*	*text*	*data*	*bss*
ML-KEM-512/FALCON1	*x5t*	In:	267556	14252	70276	87223	0	40052	26920	8	14
		Re:	265844	14252	70528	87223	0	40052	26920	8	14
	x5chain	In:	265836	14252	80428	87223	0	40052	26916	8	14
		Re:	265836	14252	80428	87223	0	40052	26916	8	14
ML-KEM-768/FALCON1	*x5t*	In:	267604	14252	72657	87339	0	40052	26904	8	14
		Re:	265844	14252	69453	87339	0	40052	26904	8	14
	x5chain	In:	267624	14252	85513	87339	0	40052	26920	8	14
		Re:	265848	14252	82181	87339	0	40052	26920	8	14
ML-KEM-512/ML-DSA-44	*x5t*	In:	251352	14252	49679	48072	0	116	27156	8	14
		Re:	249620	14252	48523	48072	0	116	27156	8	14
	x5chain	In:	251500	14252	81483	48072	0	116	27428	8	14
		Re:	249836	14252	77127	48072	0	116	27428	8	14
HQC1/FALCON1	*x5t*	In:	268580	14252	135341	87789	0	40052	27118	8	14
		Re:	260640	14252	127145	87789	0	40052	27118	8	14
BIKE1/FALCON1	*x5t*	In:	365352	14276	166026	213331	1048	40101	26944	8	14
		Re:	347380	14268	91910	213331	1048	40101	26944	8	14
ML-KEM-512/HAWK1	*x5t*	In:	266300	14252	31563	107005	0	116	26892	8	14
		Re:	264580	14252	30407	107005	0	116	26892	8	14
ML-KEM-512/HAETAE2	*x5t*	In:	255856	14772	81510	57014	520	116	26932	8	14
		Re:	254116	14772	83298	57014	520	116	26932	8	14
secp256r1/ECDSA	*x5t*	In:	224628	14256,	17151	0	0	0	26736	8	14
		Re:	224448	14256	16635	0	0	0	26736	8	14
	x5chain	In:	224708	14256	20393	0	0	0	26860	8	14
		Re:	224500	14256	19493	0	0	0	26860	8	14

cipher suite. Since both the `pqm4` and `uOSCORE-uEDHOC` implementations do not involve dynamic heap memory allocations, heap usage was not considered in this evaluation.

The overall memory usage benchmark for each cipher suite is reported in Table 6. The total size values in *bytes* for the executable program are presented in this Table, as well as the sizes corresponding to PQC libraries and the PQ-EDHOC library. Note that the defined stack size for each thread contributes to the *bss* section in the Total Size column; this stack size varies and is directly related to the specific cipher suite. Furthermore, the implementation is evaluated regarding RAM and Flash requirements, as shown in Fig. 4. Fig. 4 (left) also presents the percentage of memory used by PQC and the PQ-EDHOC library separately for FLASH values. In contrast, Fig. 4 (right) only shows the percentage of RAM used by PQC, as the RAM usage of the PQ-EDHOC library is so minimal that it is considered negligible.

Regarding **Flash usage**, since the cipher suite selection is done at compile time, the resulting code size and Flash usage can vary depending on the suit. The most significant differences are related to the varying text size requirements for the different PQC algorithms, while the PQ-EDHOC library maintains relatively consistent sizes (see Table 6). All cipher suites require a similar amount of Flash memory as the traditional secp256r1/ECDSA, except for BIKE1/FALCON1 which requires significantly more Flash, 51.5%, and 58.8% more Flash than secp256r1/ECDSA for the Responder and Initiator respectively. Even so, the Flash usage of BIKE1/FALCON1 remains only 37.9% of the total Flash available on the board.

RAM usage is a critical factor in the performance of resource-constrained embedded systems. PQC algorithms typically require more RAM compared to traditional cryptographic algorithms. This increased memory consumption may be because many available open-source implementations of PQC algorithms are not fully optimized for memory efficiency. The two categories of credential identification types in our evaluation are *x5t* and *x5chain*.

The *x5chain* generally requires more RAM than *x5t*, as it necessitates larger buffer sizes associated with the TX messages to accommodate the full X.509 certificate included in the *x5chain* credential type, whereas *x5t* only contains the certificate's identification value. Specifically for the Responder, approximately 9.3% more RAM in *x5chain* is needed compared to *x5t* for secp256r1/ECDSA, 11.7% more for ML-KEM-512/FALCON1, 15.2% more for ML-KEM-768/FALCON1 and 45.5% more for ML-KEM-512/ML-DSA-44. For the Initiator, 10.3% more RAM in *x5chain* is needed compared to *x5t* for secp256r1/ECDSA, 12% more for ML-KEM-512/FALCON1, 14.8% more for ML-KEM-768/FALCON1 and 49.8% more for ML-KEM-512/ML-DSA-44.

In the case of *x5t*, the results can be grouped into three groups: Firstly, the RAM usage increases of HQC1/FALCON1 and BIKE1/FALCON1 compared to secp256r1/ECDSA range from 243% to 474%, with Initiator requiring more memory due to the high memory usage of the Key Generation operation. Secondly, the RAM usage increases of ML-KEM-512/FALCON1, ML-KEM-768/ FALCON1 and ML-KEM-512/HAETAE2 range from 173% to 217%. In this case, Responder and Initiator

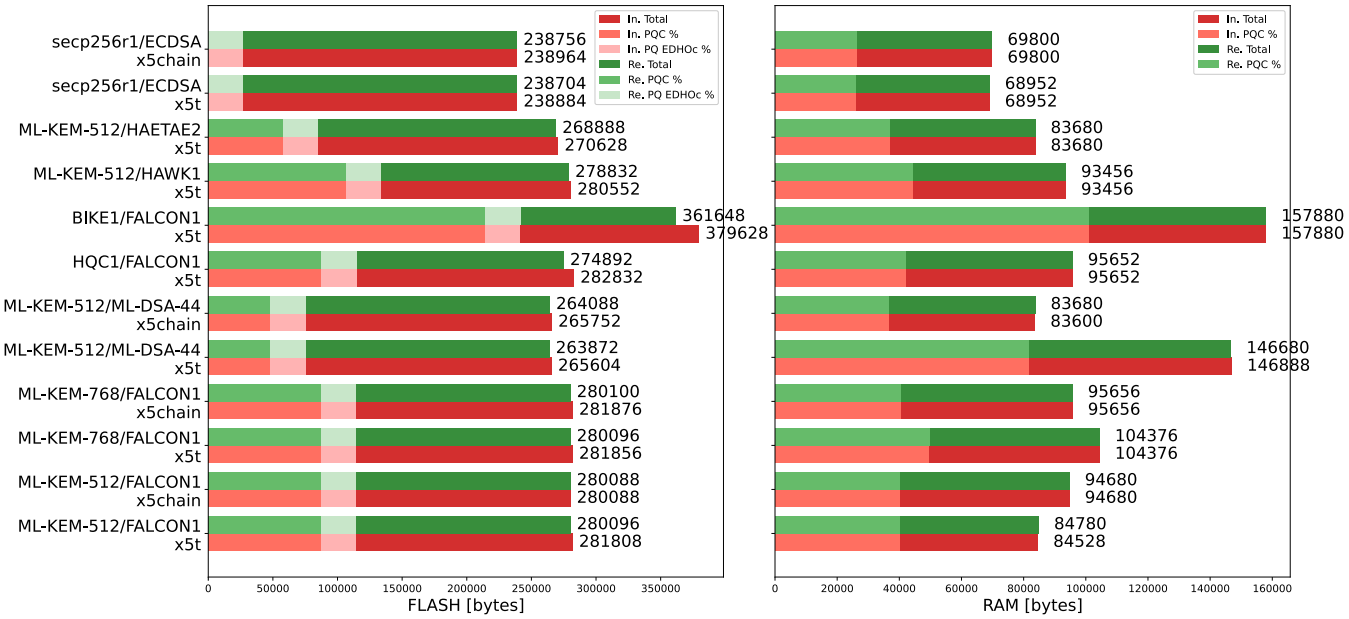


FIGURE 4. PQ-EDHOC FLASH and RAM for initiator an responder.

TABLE 7. PQ-EDHOC stack for initiator and responder.

Cipher Suite	Cred. Type	Stack Size (Bytes)	
		Initiator	*Responder*
ML-KEM-512/FALCON1	*x5t*	22248	21072
	x5chain	33344	30776
ML-KEM-768/FALCON1	*x5t*	24360	22480
	x5chain	35456	32184
ML-KEM-512/ML-DSA-44	*x5t*	39760	38648
	x5chain	67696	63392
HQC1/FALCON1	*x5t*	83572	75380
BIKE1/FALCON1	*x5t*	117104	43064
ML-KEM-512/HAWK1	*x5t*	23544	22440
ML-KEM-512/HAETAE2	*x5t*	72580	74348
secp256r1/ECDSA	*x5t*	10480	10040
	x5chain	13416	12608

have smaller differences due to the smaller cost of Key Generation of ML-KEM. Finally, ML-KEM-512/ML-DSA-44 and ML-KEM-512/HAWK1 require more acceptable memory consumption, with usage increases of 103% and 45%, respectively.

Falcon's implementation in `pqm4` requires a lot of *bss* memory, contributing to an increase in the total RAM requirement of the cipher suites that include it. This is evident in Fig. 4, where the percentage of PQC in the total RAM requirement for cipher suites with Falcon, ranges from 22.8% to 47.8%. In contrast, for the other cipher suites, the PQC memory percentage is below 1%.

Regarding RAM consumption, we can see that all PQ cipher suites require more memory than the traditional ECDH and ECDSA cipher suites. Specifically, for cipher suites using BIKE1 and HQC1 as the KEM, it can be observed that the memory consumption is significantly larger than traditional PKC cipher suites, using up to 70% and 58% respectively from the total available RAM, in the case of Initiator with *x5t*. This may become prohibitive in resource-constrained devices like the one we evaluate when combined with a real application. This can be seen in the real-world benchmark in Section VI-C as the memory requirement prohibited us from evaluating the above-mentioned PQC cipher suites. On the other hand, for cipher suites using ML-KEM-512 in combination with HAWK1, ML-DSA-44, and FALCON1, the memory consumption is acceptable, using up to 18%, 25%, and 33% respectively from the total available RAM, in the case of Initiator with *x5t*.

C. REAL-WORLD ANALYSIS AND EVALUATION OF PQ-EDHOC

To evaluate the protocol in a real-world IoT scenario in terms of both execution time and energy requirements, a 6LowPAN (IPv6 over BLE) network was established consisting of an nRF52840DK BLE SoC and a Linux Workstation. The 6LoWPAN IPv6 over BLE stack was chosen to represent a low-power IoT communication setup suitable for evaluating the feasibility of PQ-EDHOC in battery-powered devices. BLE transmission power is set to 0 dBm, with a maximum data length of 251 bytes per BLE packet (the limit in Bluetooth 4.2) and a Logical Link Control and Adaptation Protocol (L2CAP) Maximum Transmission Unit (MTU) of 512 bytes. Both devices connect directly to each other, without any intermediate gateways, achieving a mean Round-Trip Time (RTT) of 77.77 ms across 100 interactions.

Zephyr OS [69] (Version 3.4.0) was used on the micro-controller. The network stack is composed of PQ-EDHOC

over CoAP with block-wise transfer at the application level, User Datagram Protocol (UDP) at the transport layer, and Internet Protocol Support Profile (IPSP) in the Node role, using 6LoWPAN providing IPv6 connectivity over BLE. For block-wise CoAP transfer on Zephyr, we employed the *Client-Coap API* library in the embedded system and the `libcoap` [62] library on the Linux workstation.

The embedded system runs the Initiator and the Linux workstation runs the Responder with the block-wise transfer block size set to 512 bytes on both sides. The evaluation is made under the ideal environment where the BLE devices are close to each other and there is no packet loss; the effect of network packet loss in PQ-EDHOC is studied in Section VII. The PQ-EDHOC solution is evaluated in the real-world environment by assessing the Initiator performance on the embedded device and comparing it with the traditional EDHOC. The total handshake for the Initiator is measured from the ephemeral key generation until the OSCORE context derivation. Note that for the real-world IoT scenario, the HQC1 and BIKE1 KEMs are excluded because their memory requirements exceed the available memory of the platform when the network stack is enabled. Also for this experiment, cipher suites only with the *x5t* credential type are compared, to reduce the message sizes.

We separately analyze the execution process, and the network transmission and reception processes for execution time and average power consumption. Specifically, the transmission process (`Tx`) starts when EDHOC has the first message ready and ends when the last CoAP Confirmable Block is sent. The reception process (`Rx`) is measured from the point that the first `ACK 2.04 Changed` message is received until the last `ACK 2.31 Continue` message is received.

In Table 8 and Fig. 5 (left), the total handshake time, processing time, and `Tx/Rx` Time are reported, as the average of 10 interactions. The power and energy evaluation are also presented in Table 9 and 5 (right). The total memory usage in terms of *text*, *data*, and *bss* sizes, along with the total communication transmission size and the percentage of the available memory that is being used for RAM and FLASH, is reported in Table 10.

1) SPEED EVALUATION

It can observed from Tables 8 and 10 that there are two cipher suites performing better than the traditional secp256r1/ECDSA, namely ML-KEM-512/FALCON1 and ML-KEM-512/HAWK1, which are 19% and 54% faster compared to secp256r1/ECDSA respectively. The other two cipher suites, ML-KEM-512/ML-DSA-44 and ML-KEM-512/HAETAE2 perform worse than secp256r1/ECDSA. We also observe that all PQ cipher suites have a faster processing time than secp256r1/ECDSA. For ML-KEM-512/ML-DSA-44 and ML-KEM-512/HAETAE2, the poorer performance can be attributed to the transmission and reception time (`Tx/Rx`). The larger sizes of artifacts that the PQC algorithms introduce as they are presented in Table 10

TABLE 8. PQ-EDHOC total handshake time.

Cipher Suite	Cred. Type	Total Handshake Time (ms)	Processing Time (ms)	Tx/Rx Time (ms)
ML-KEM-512/FALCON1	*x5t*	1453.36	764.84	688.52
ML-KEM-512/ML-DSA-44	*x5t*	3103.93	568.86	2535.07
ML-KEM-512/HAWK1	*x5t*	825.43	133.58	691.85
ML-KEM-512/HAETAE2	*x5t*	2370.32	525.83	1844.49
secp256r1/ECDSA	*x5t*	1748.81	1431.26	251.19

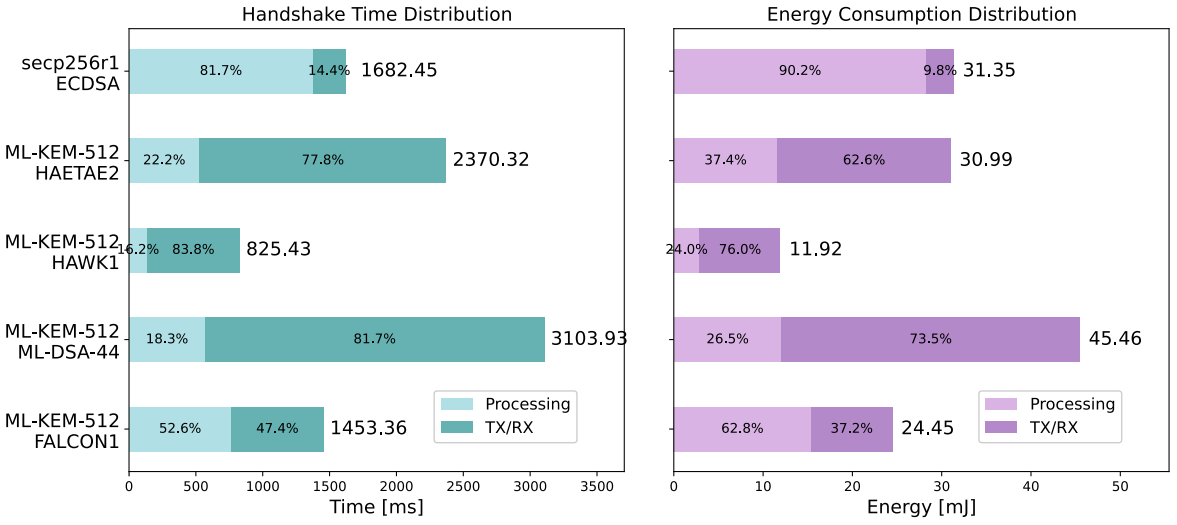


FIGURE 5. PQ-EDHOC handshake time and energy consumption.

TABLE 9. PQ-EDHOC total handshake power and energy evaluation.

Cipher Suite	Cred. Type	Avg. Power (mW)			Energy (mJ)		
		Total	Proc.	Tx/Rx	Total	Proc.	Tx/Rx
ML-KEM-512/FALCON1	*x5t*	16.82	20.08	13.21	24.45	15.35	9.09
ML-KEM-512/ML-DSA-44	*x5t*	15.96	21.16	13.18	45.46	12.03	33.42
ML-KEM-512/HAWK1	*x5t*	14.46	21.45	13.1	11.92	2.86	9.06
ML-KEM-512/HAETAE2	*x5t*	16.53	22.06	10.51	30.99	11.59	19.39
secp256r1/ECDSA	*x5t*	18.40	19.76	12.23	31.35	28.28	3.07

TABLE 10. PQ-EDHOC evaluation of communication size and memory usage.

Cipher Suite	Cred. Type	Comm. Sizes (Bytes)	Memory Usage (Bytes)			% FLASH Usage (%)	% RAM Usage (%)
			text	*data*	*bss*		
ML-KEM-512/FALCON1	*x5t*	3959	517998	21148	153075	53.9	68.1
ML-KEM-512/ML-DSA-44	*x5t*	9780	499862	21100	147295	52.1	65.8
ML-KEM-512/HAWK1	*x5t*	3756	516878	21148	113139	53.8	52.5
ML-KEM-512/HAETAE2	*x5t*	7208	504930	21564	157255	52.6	69.9
secp256r1/ECDSA	*x5t*	572	468230	21148	97671	48.9	46.4

impose a significant overhead in the total handshake time when the packets are transmitted and received by the BLE protocol. This highlights the importance of PQC algorithms with smaller communication sizes in resource-constrained network applications.

2) POWER AND ENERGY EVALUATION

Regarding power and energy consumption, it can be observed that the average power consumption during the processing part is higher than during the transmission and reception phases of the protocol. This is the primary reason we observe three cipher suites consuming less energy than the traditional secp256r1/ECDSA. Specifically, while ML-KEM-512/HAETAE2 requires more total time for a single handshake than secp256r1/ECDSA, the majority of the time is spent in the Tx/Rx phase, which consumes less Power and, consequently, less Energy. As a result, the total energy consumption is slightly less than secp256r1/ECDSA. On the other hand, ML-KEM-512/ML-DSA-44 does not achieve better energy consumption but still performs better in the total handshake time when compared to secp256r1/ECDSA.

To evaluate if the proposed PQ-EDHOC protocol can be prohibitive in some battery-powered applications, we consider an example with a CR2032 coin cell battery with a capacity of 2322 J. It can now be measured how many interactions of the PQ-EDHOC protocol, with different cipher suites, can be executed in such a setup. The best performing ML-KEM-512/HAWK1 can be executed ≈194700 times and the worst ML-KEM-512/ML-DSA-44 can be executed ≈51000 times, while secp256r1/ECDSA can be executed ≈74000 times. It can be concluded that PQ-EDHOC, not only is suitable for real-world battery-powered IoT devices but can be more performative with more energy-friendly cipher suites compared to traditional EDHOC
solutions.

3) MEMORY USAGE EVALUATION

Regarding Flash usage, as shown in Fig. 6 (left) PQ-EDHOC requires slightly more memory than traditional EDHOC. EDHOC with cipher suite secp256r1/ECDSA utilizes 48.9% of total Flash available on our embedded system. However, PQ-EDHOC, in the worst case, requires 53.9%. This suggests that, concerning Flash usage, all cipher suites are compatible with our embedded system.

Regarding RAM usage, Fig. 6 (right) shows that PQ-EDHOC requires more memory than traditional EDHOC. Traditional secp256r1/ECDSA requires 46.4% of the total RAM of the embedded system, while PQ-EDHOC with cipher suite ML-KEM-512/HAWK1 requires only 13% more RAM, making it an appealing choice for real-world IoT devices. The rest of the cipher suites need more than 66% of RAM to operate, putting more constraints on the available memory for a real-world application that will operate on such a system.

4) CoAP OVERHEAD EVALUATION

The overhead imposed by CoAP can be found by computing the size of the artifacts that need to be sent before CoAP and the actual message size that is sent "over-the-wire" with CoAP. Results are reported in Table 4 and Table 10 respectively. For secp256r1/ECDSA, the CoAP overhead is 330 bytes, 35% more bytes than the messages we wanted to send. The overheads are 1029 bytes (51.3%) for ML-KEM-512/FALCON1, 3317 bytes (37.4%) for ML-KEM-512/ML-DSA-44, 1023 bytes (57%) for ML-KEM-512/HAWK1, and for ML-KEM-512/HAETAE2 2637 bytes (42%). It can be observed that the larger the message, the larger the CoAP overhead is.

VII. NETWORK OVERHEAD ANALYSIS OF PQ-EDHOC

A. EXPERIMENT SETUP

To evaluate the performance of the PQ-EDHOC over different network conditions, we measure the handshake

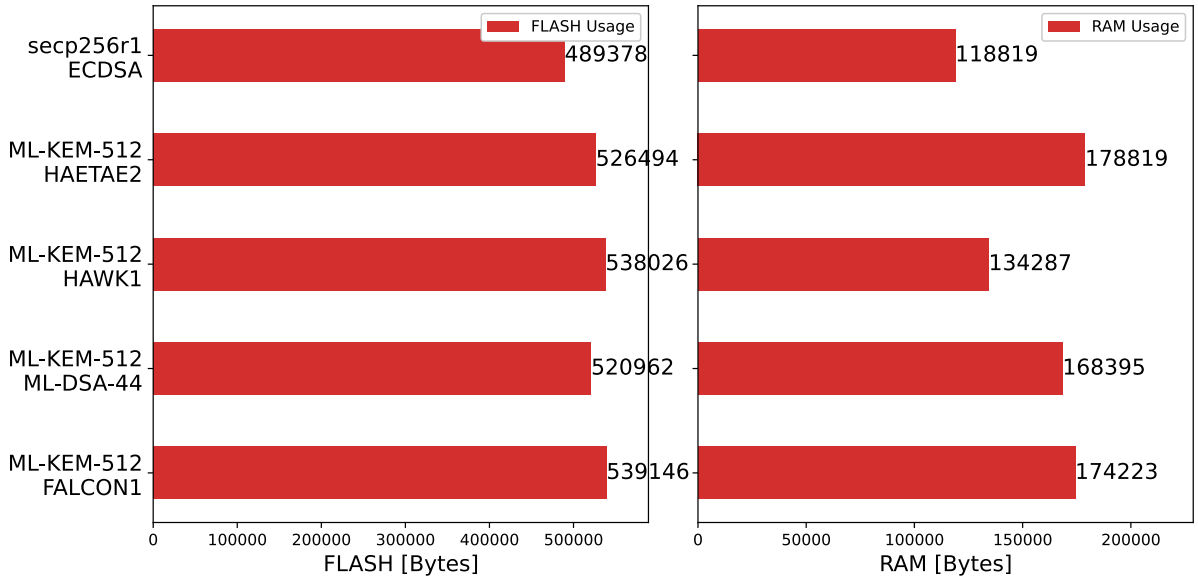


FIGURE 6. PQ-EDHOC memory size (FLASH and RAM).

time under different network latencies and packet loss rates. A benchmarking methodology is adapted similarly to the benchmarking of post-quantum TLS protocol by Paquin et al. [12].

The benchmark platform has an AMD Ryzen 5 7640HS CPU at 4.30 GHz and 32 GB memory. 3 VirtualBox virtual machines (VMs) were set up for the benchmark: a client VM, a server VM, and a router VM connecting them. We use the `netem` to emulate different network latencies and packet loss rates.

For the network latency, two communication scenarios between data centers within the same country were utilized, as used in benchmarking post-quantum TLS to emulate different network distances: the target RTT in these four scenarios are approximately 6.193 ms and 30.906 ms respectively. The baseline RTT of our VM network is about 0.863 ms. We add $((Target\_RTT - Baseline\_RTT)/2)$ milliseconds delay to each outgoing packet on both network interfaces in the router VM via `netem`. Due to the small extra latency introduced by `netem`, the actual measured RTTs in our emulation are 6.292 ms and 31.201 ms respectively.

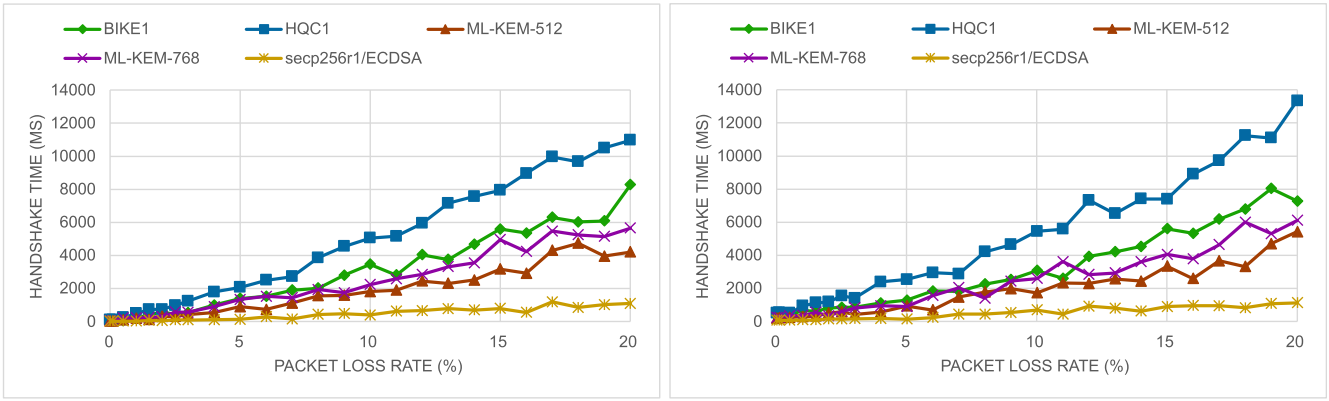
Because the current `libcoap` implementation can only perform the retransmissions on confirmable messages i.e. the EDHOC requests, we apply the packet loss rate on the network interface of the client VM via `netem` instead. Paquin et al. discussed the packet loss rate on the Internet, noting that 97% of Internet traffic has packet loss of less than 20%, based on the telemetry collected by Mozilla in 2019. Accordingly, a drop between 0% and 20% of the outgoing packets randomly on the network interface was chosen in the client VM via `netem` in our emulation.

The handshake time between the ephemeral key generation and the OSCORE context derivation on the Initiator with *x5t* was measured in this experiment. We use the UDP as the transport layer protocol. We use the same block size of 512 bytes as in the previous experiments has been used and the maximum number of retransmissions was increased to 10 for the CoAP Block-Wise Transfer. The default values for all the other CoAP parameters [46] were used. We collect 100 measurements for each combination of cipher suite, network latency, and packet loss rate. Because the delays due to CoAP retransmissions are much higher (in the order of seconds) than the RTT and the cryptographic operations, the collected measurements will contain outliers (very large latency compared to the other measurements under the same network condition) when the packet loss rate is high. Thus, we calculate the trimmed means within the two standard deviations (95%) for the measurements to discard the outliers.

B. COMPARISON BETWEEN DIFFERENT KEMS

We measure the (trimmed) mean handshake time of PQ-EDHOC with different KEMs under various network conditions, combined with the same digital signature algorithm FALCON1. Results are shown in Fig. 7. Note that the secp256r1/ECDSA is the classical Diffie-Hellman without PQC and it is presented in the figures for comparison purposes.

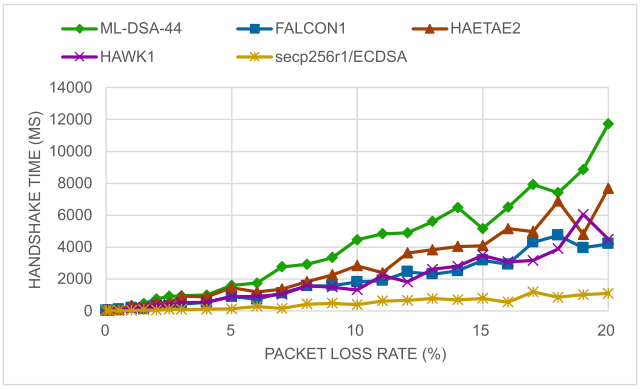
Fig. 7 shows the trend that when the packet loss rate increases, the cipher suites with smaller KEM (see Table 3) will outperform the cipher suites with larger KEM for both RTTs. For larger packet loss rates, we have ML-KEM-512 > ML-KEM-768 > BIKE1 > HQC1 when combining with FALCON1 in PQ-EDHOC (“>” reads as “outperform”). In addition, the classical secp256r1/ECDSA significantly outperforms all the PQC cipher suites for large packet loss rates in this experiment. Note that this is different from the results in Section VI, where the cipher suite ML-KEM-512/FALCON1 achieves better handshake time than



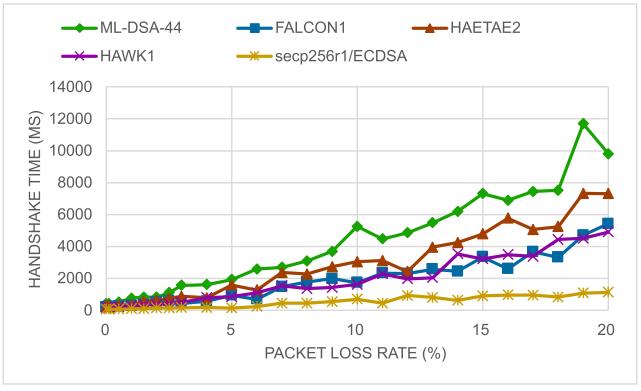
(a) RTT=6.292ms

(b) RTT=31.201ms

FIGURE 7. Handshake time of different KEMs with FALCON1.



(a) RTT=6.292ms



(b) RTT=31.201ms

FIGURE 8. Handshake time of different signatures with ML-KEM-512.

secp256r1/ECDSA. It is mainly because secp256r1/ECDSA has much larger computation overhead than ML-KEM-512/FALCON1 (see Table 8 and Fig. 5). Under good network conditions with very low packet loss rates (e.g. the experiment environment in Section VI), the computation time will be more dominant in the handshake time. However, when the packet loss rate becomes higher, the (re)transmission time will dominate, because each retransmission will add a delay in the order of seconds, which is much bigger than the computation time.

In addition, because the retransmission latencies are also much larger than the RTTs used in this experiment, the mean handshake times for the same cipher suite under the same packet loss rate are similar i.e., the RTT does not have a significant impact on the handshake time in this experiment except for the networks with very low packet loss rates.

C. COMPARISON BETWEEN DIFFERENT SIGNATURES

Here, we also measure the (trimmed) mean handshake time of PQ-EDHOC with different digital signatures under various network conditions, combined with the same KEM algorithm ML-KEM-512. Results are shown in Fig. 8.

Similar to the results with different KEMs, Fig. 8 shows the trend that when the packet loss rate increases, the cipher suites with more compact digital signatures in terms of the communication size (see Table 1) will outperform the cipher suites with less compact digital signatures for both RTTs. For larger packet loss rates, we have FALCON1/HAWK1 > HAETAE2 > ML-DSA-44 when combined with ML-KEM-512 in PQ-EDHOC. FALCON1 and HAWK1 achieve similar performance due to their similar communication sizes. Note that this is also different from the results in Section VI, where the cipher suite ML-KEM-512/HAWK1 achieves better handshake time than ML-KEM-512/FALCON1. It is because ML-KEM-512/HAWK1 has much lower computation overhead than ML-KEM-512/FALCON1 (see Table 8 and Fig. 5) and can be explained by the similar rationales as in Section VII-B. In addition, the RTT also does not have a significant impact on the handshake time in this experiment except for the networks with low packet loss rates, due to the same reasons explained in Section VII-B.

VIII. CONCLUSION

The benchmark results show that a quantum-resistant version of EDHOC with the selected cipher suites evaluated in

a real-world IoT scenario is not prohibitive. Furthermore, two cipher suites, ML-KEM-512/HAWK1, and ML-KEM-512/FALCON1, could be considered suitable replacements for the elliptic curve cryptographic primitives. Both perform better regarding execution time and energy consumption with an overhead in memory consumption. As research progresses, we can expect more optimized implementation that will further reduce memory costs. Finally, EDHOC running with ML-KEM-512/HAWK1 cipher suite is a promising candidate, even with the current unoptimized implementation of HAWK1. However, its success will depend on the standardization of HAWK signatures through the NIST process. In the evaluated real-world IoT scenario, the overhead imposed by the larger artifacts of the PQC algorithms plays a crucial role in the total handshake time. On the other hand, this cost is not so strongly reflected in the energy consumption, due to the use of hardware-supported low-power transmission protocols. An analysis and evaluation of different transmission protocols for 6LoWPAN in real-world IoT scenarios could be valuable in assessing the actual cost of the quantum-resistant transition of EDHOC. Additionally, there is a gap in optimized implementations for even more constrained devices, such as Cortex-M0, which also would be a relevant target system for EDHOC. An evaluation of a real-world scenario in which EDHOC uses the *x5chain* COSE credential type, where a chain of PQ certificates are transmitted, is crucial to cover all the possible use cases of EDHOC. We can expect a stronger impact on the transmission cost, leading to a larger difference in handshake times for algorithms with large artifacts, such as ML-DSA-44.

APPENDIX A
MEASUREMENT PROCESS

A. SPEED BENCHMARKING

The built-in timer on the nRF52840 chip [73] was used to record cycle counts for timing analysis. This timer is a general-purpose peripheral that operates on the High-Frequency clock (HFCLK) with a base frequency of 16 MHz. The external high-frequency crystal oscillator (HFXO) was enabled to improve timing accuracy. The cycle counts captured by the 16 MHz timer are then scaled to the 64 MHz CPU clock to estimate the CPU cycle count. 100 executions per scheme were performed and the average, maximum, and minimum values were listed.

B. POWER AND ENERGY BENCHMARKING

Power and energy consumption were measured on the different tests by monitoring the current drawn by the nrF52840 SOC through the nrF52840DK. To take full advantage of the nRF52840 low-power modes, the chip was set to operate at the lower power voltage VDD (1.7 V to 3.6 V), and to avoid potential noise from the USB power supply, the DK was powered from an external power supply on connector P21 with 3.0 V. The DK was modified for power measurement by cutting the PCB track, shorting the solder bridge, and connecting a 10Ω resistor. The instantaneous current was calculated from the voltage drop across the resistor by measuring the difference of the voltages measured at two probes in an oscilloscope (Pico-Scope 5444B from the PicoScope 5000 Series [74] with 1 GS/s real-time sampling). Three trigger signals are connected to the oscilloscope from the GPIO D0, D1, and D3 board pins to control the power ON/OFF measured by different primitives on every test. Note that probes with 1x attenuation were used and the oscilloscope averaging mode was enabled to reduce noise. The sampling rate was set to 2 MS, 14-bit hardware resolution, and a repeated external trigger was used to get as many interactions of the algorithms as can be kept in the oscilloscope memory buffer. The gathered measurements were inputted into a Python script to compute the average power consumption, CPU processing time, and total energy consumption.

The 64 MHz High-Frequency Clock (HFCLK), sourced either from the High-Frequency Internal Clock (HFINT) or from the 32 MHz High-Frequency Crystal Oscillator (HFXO), is only enabled in the nrf52840 when necessary, to achieve significant power saving. In contrast, the Low-Frequency Clock (LFCLK) clock, sourced either from the Low-Frequency Resistor-Capacitor Oscillator (LFRC) or from the Low-Frequency Crystal Oscillator (LFXO), operates continuously. To enhance power efficiency during power consumption measurements on both PQC primitives and PQ-EDHOC total handshake tests, the HFCLK was disabled and the LFCLK was configured to use the LFRC oscillator in its ultra-low power (ULP) mode. By default, the nRF52840 employs the LFXO as the LFCLK source when the Bluetooth Low Energy (BLE) stack is initialized, due to its higher accuracy than the LFRC oscillator. This precision is critical for BLE operations, including advertising and maintaining connection intervals for reliable communication. Therefore, it is essential to retain this specific configuration when power consumption measurements are taken in the real-world scenario where the test is running in the IPv6 over the BLE network.

C. OVERALL CODE SIZE AND MEMORY USAGE

The overall code size has been calculated in terms of the *text*, *bss*, and *data* sections and reported separately by using the `arm-zephyr-eabi-size` tool from ARM GCC toolchain. Additionally, the total Flash and RAM usage has been reported separately.

APPENDIX B
GLOSSARY OF TERMS

A. CRYPTOGRAPHIC TERMS

TRADITIONAL/CLASSICAL CRYPTOGRAPHY: Refers to public-key cryptographic systems that rely on algorithms based on mathematical problems such as integer factorization, finite-field discrete logarithms, or elliptic-curve

discrete logarithms [8] to achieve a particular cryptographic outcome. Common examples include RSA (Rivest-Shamir-Adleman), Diffie-Hellman (DH), Elliptic Curve Diffie-Hellman (ECDH), and Elliptic Curve Digital Signature Algorithm (ECDSA), which are widely used in secure communications today.

PQC (POST-QUANTUM CRYPTOGRAPHIC): Refers to cryptography systems or schemes that rely on algorithms specifically designed to remain secure even in the presence of large-scale quantum computers, which can break Traditional/Classical Cryptographic systems. **PQC algorithms** are intended to be resistant to attacks by quantum computers, which use quantum-mechanical phenomena to solve mathematical problems that are infeasible for classical computers.

SHOR'S ALGORITHM: A quantum algorithm developed by Peter Shor [1] that efficiently solves the integer factorization problem and the discrete logarithm problem—two mathematical problems that form the security foundation of most public-key cryptographic schemes, including RSA, Diffie-Hellman, and ECC. By leveraging the principles of quantum computing, Shor's algorithm may break these cryptographic systems in polynomial time, posing a significant threat to classical encryption methods.

GROVER'S ALGORITHM: A quantum algorithm developed by Lov K. Grover that optimally solves the general search problem by providing a quadratic speedup over classical brute-force search. By leveraging the principles of quantum computing, Grover's algorithm may reduce the effective security strength of symmetric cryptographic systems, such as AES and SHA, by approximately half.

KEY AGREEMENT AND KEY TRANSPORT: Refer to schemes used to establish a shared cryptographic key for secure communication. Common mechanisms employed include Diffie-Hellman (DH) or Elliptic Curve Diffie-Hellman (ECDH), which rely on public key cryptography and are vulnerable to Shor's algorithm [8]. As a result, these mechanisms are among those that need to be replaced by PQC solutions.

DIGITAL SIGNATURES: Refer to schemes used to authenticate the identity of a sender, detect unauthorized modifications to data, and underpin trust in a system. Traditional/classical signatures also rely on a public-private key pair, so a breach in public key cryptography would also compromise them [8]. As a result, post-quantum digital signatures should be developed to replace the classical ones.

KEM: A *key-encapsulation mechanism* is a cryptographic scheme consisting of three functions: Key Generation, Encapsulation, and Decapsulation. Under certain conditions, it can be used by two parties to establish a shared secret key over a public channel [2], [7].

SECURITY CLASSIFICATION OF PQC ALGORITHMS: NIST specified five security strength categories:
Level 1: equivalent to AES-128 key search;
Level 2: equivalent to SHA-256/SHA3-256 collision search;
Level 3: equivalent to AES-192 key search;
Level 4: equivalent to SHA-384/SHA3-384 collision search;
Level 5: equivalent to AES-256 key search.

B. NIST PQC STANDARDIZED ALGORITHMS

NIST has standardized three PQC algorithms: one PQC KEM (ML-KEM) and two PQC Digital Signatures (ML-DSA and SLH-DSA). Furthermore, the FALCON PQC digital signature is right now in the process of standardization as FN-DSA.

ML-KEM *Module-Lattice-based Key-Encapsulation Mechanism Standard (FIPS-203) [2]*: This standard specifies a KEM based on the CRYSTALS-Kyber algorithm, which has been renamed ML-KEM. Its security is related to the computational difficulty of the Module Learning with Errors problem. This standard specifies three parameter sets, in order of increasing security strength and decreasing performance: ML-KEM-512 (Security Level 1), ML-KEM-768 (Security Level 3), and ML-KEM-1024 (Security Level 5).

ML-DSA *Module-Lattice-Based Digital Signature Standard (FIPS-204) [3]*: This standard specifies a set of algorithms intended as the primary standard for protecting digital signatures. The standard uses the CRYSTALS-Dilithium algorithm, which has been renamed ML-DSA and can be used to generate and verify PQC digital signatures. The ML-DSA standard also specifies parameter sets, namely ML-DSA-44 (Security Level 2), ML-DSA-65 (Security Level 3), and ML-DSA-87 (Security Level 5).

SLH-DSA *Stateless Hash-Based Digital Signature (FIPS-205) [4]*: This standard employs the Sphincs+ algorithm, which has been renamed SLH-DSA. It is designed for digital signatures, and its security relies on the presumed difficulty of finding preimages for hash functions as well as several related properties of the same hash functions. Based on a different mathematical approach than ML-DSA, it serves as a backup method in case ML-DSA proves vulnerable. The SLH-DSA standard also specifies parameter sets, namely SLH-DSA-128 (Security Level 1), SLH-DSA-192 (Security Level 3), and SLH-DSA-256 (Security Level 5).

FN-DSA *FALCON lattice signature scheme [55]*: This algorithm is also a lattice-based signature scheme, with its underlying hard problem being the Short Integer Solution problem (SIS) over NTRU lattices. By utilizing NTRU lattices, the scheme achieves substantially shorter signatures than other lattice-based signature schemes. Additionally, the use of fast Fourier sampling allows for highly efficient and

fast implementations. FALCON1 (Security Level 1) specifies the lowest security and highest performance parameter set of the algorithm.

C. FOURTH ROUND NIST CANDIDATES

The fourth round of the NIST process aims to select an alternative algorithm for KEM that is based on a different hard problem than ML-KEM. The candidates still advancing for standardization are:

BIKE [50]: A KEM based on the difficulty of decoding Quasi-Cyclic Moderate Density Parity Check (QC-MDPC) codes, which are used for error correction with efficient bit-flipping techniques. BIKE1 (Security Level 1) corresponds to the parameter set with the lowest security level and highest performance (BIKE-80).

HQC: A KEM based on the hardness of decoding Quasi-Cyclic Concatenated Reed Muller Reed Solomon (RMRS) codes. These codes are used in error correction, especially in communication systems like wireless and space communications. HQC1 (Security Level 1) corresponds to the parameter set with the lowest security level and highest performance (HQC-128).

ClassicMcEliece [51]: A KEM based on the difficulty of decoding Goppa codes, a class of error-correcting codes that help recover messages from corrupted data during transmission.

D. ADDITIONAL PQC DIGITAL SIGNATURE

NIST has selected 14 candidates for the second round of the Additional Digital Signatures for the NIST PQC Standardization Process. The advancing digital signature algorithms include: CROSS FAEST, HAWK, LESS, MAYO, Mirath (merger of MIRA/MiRitH), MQOM, PERK QR-UOV and RYDE. It is important to note that while the HAETAE algorithm, included in the comparison study of this paper, was part of the call for digital signatures, it did not progress to NIST round 2. However, HAETAE has recently been selected for standardization by the Korean PQC process [59].

HAWK [57]: A signature scheme based on the Lattice Isomorphism Problem (LIP), focusing on simplicity. Using module lattices inspired by NTRUSign and Falcon, HAWK eliminates the need for floating-point operations and rejection sampling, while utilizing compact precomputed distributions, making it ideal for constrained devices. According to its authors [57], HAWK outperforms Falcon in several aspects: Hawk-512 (HAWK1 with Security Level 1) and Hawk-1024 are four times faster than Falcon on x86 architectures, produce 15% more compact signatures, and offer slightly stronger security against lattice reduction attacks. Additionally, when floating-point operations are unavailable, Hawk is 15 times faster than Falcon.

HAETAE *(Hyperball bimodal module rejection signature scheme)*: A lattice-based signature scheme designed to improve the complexity/compactness trade-off for resource-constrained environments. Like the NIST-selected ML-DSA, HAETAE uses the Fiat-Shamir with Aborts paradigm but focuses on reducing signature and verification key sizes. According to their authors [58], its signatures and verification key sizes of HAETAE2 (Security Level 2) are up to 39% and 25% smaller than ML-DSA-768 while maintaining high security against various attacks. Additionally, HAETAE is resistant to implementation attacks, such as side-channel analysis, making it well-suited for IoT and embedded systems.

APPENDIX C
ABBREVIATIONS AND ACRONYMS

Abbreviation	Definition
RSA	Rivest-Shamir-Adleman Algorithm.
ECC	Elliptic Curve Cryptography.
AES	Advanced Encryption Standard.
SHA	Secure Hash Algorithm.
MAC	Message Authentication Code.
AES-CCM	AES Counter with CBC-MAC.
PRK	Uniformly pseudo-random keys.
PKC	Public Key Cryptography.
PQC	Post-Quantum Cryptography.
NIST	National Institute of Standards and Technology
FIPS	Federal Information Processing Standards
KEM	Key Encapsulation Mechanism.
ML-KEM	module-lattice-based key encapsulation mechanism
ML-DSA	module-lattice-based digital signature algorithm.
SLH-DSA	stateless hash-based digital signature algorithm.
PQUIC	Post-Quantum Use in Protocols.
TLS 1.3	Transport Layer Security version 1.3.
EDHOC	Ephemeral Diffie-Hellman Over COSE.
MQTT	Message Queue Telemetry Transport
TLS	Transport Layer Security.
DTLS	Datagram TLS
QUIC	Quick UDP Internet Connections.
OSCORE	Object Security for Constrained RESTful Environments.
HKDF	HMAC-based Extract-and-Expand Key Derivation Function.
CBOR	Concise Binary Object Representation.
COSE	CBOR (Concise Binary Object Representation) Object Signing and Encryption.
X.509	Public Key Infrastructure Certificate.
c509	CBOR Encoded X.509 Certificates.
CoAP	Constrained Application Protocol.

CWT	CBOR Web Token.
CCS	CWT Claims Set.
IANA	Internet Assigned Numbers Authority.
RTOS	Real-time Operating System.
BLE	Bluetooth Low Energy.
6LoWPAN	IPv6 over Low Power Wireless Personal Area. Networks.
L2CAP	Logical Link Control and Adaptation Protocol.
MTU	Maximum Transmission Unit
RTT	Round-Trip Time.
UDP	User Datagram Protocol.
IPSP	Internet Protocol Support Profile.

REFERENCES

[1] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134.

[2] NIST. (2024). *FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard*. [Online]. Available: https://csrc.nist.gov/pubs/fips/203/final

[3] NIST. (2024). *FIPS 204: Stateless Hash-Based Digital Signature Standard*. [Online]. Available: https://csrc.nist.gov/pubs/fips/204/final

[4] NIST. (2024). *FIPS 205: Module-Lattice-Based Key-Encapsulation Mechanism Standard*. [Online]. Available: https://csrc.nist.gov/pubs/fips/205/final

[5] NIST. (2023). *Post-Quantum Cryptography: Digital Signature Schemes*. [Online]. Available: https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures

[6] *Status Report on the First Round of the Additional Digital Signature Schemes for the Nist Post-quantum Cryptography Standardization Process*, document IR 8528, NIST, Washington, DC, USA, 2024.

[7] F. Driscoll, M. Parsons, and B. Hale. (Sep. 2024). *Terminology for Post-Quantum Traditional Hybrid Schemes*. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-pquip-pqt-hybrid-terminology/04/

[8] A. Banerjee, T. Reddy, D. Schoinianakis, T. Hollebeek, and M. Ounsworth. (Oct. 2024). *Post-Quantum Cryptography for Engineers*. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-pquip-pqc-engineers/06/

[9] N. Bindel, B. Hale, D. Connolly, and F. Driscoll. (Nov. 2024). *Hybrid Signature Spectrums*. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-pquip-hybrid-signature-spectrums/03/

[10] 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," *Nature*, vol. 605, no. 7909, pp. 237–243, May 2022.

[11] H. Nejatollahi, N. Dutt, S. Ray, F. Regazzoni, I. Banerjee, and R. Cammarota, "Post-quantum lattice-based cryptography implementations: A survey," *ACM Comput. Surveys*, vol. 51, no. 6, pp. 1–41, Jan. 2019.

[12] C. Paquin, D. Stebila, and G. Tamvada, "Benchmarking post-quantum cryptography in TLS," in *Proc. Int. Conf. Post-Quantum Cryptogr.* Cham, Switzerland: Springer, Jan. 2020, pp. 72–91.

[13] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis, "Post-quantum authentication in TLS 1.3: A performance study," presented at the 27th Annu. Netw. Distrib. Syst. Secur. (NDSS) Symp., San Diego, CA, USA, 2020. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2020/02/24203-paper.pdf

[14] K. Bürstinghaus-Steinbach, C. Krauß, R. Niederhagen, and M. Schneider, "Post-quantum TLS on embedded systems: Integrating and evaluating kyber and SPHINCS+ with mbed TLS," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 841–852.

[15] G. Tasopoulos, J. Li, A. P. Fournaris, R. K. Zhao, A. Sakzad, and R. Steinfeld, "Performance evaluation of post-quantum tls 1.3 on resource-constrained embedded systems," in *Information Security Practice and Experience*. Cham, Switzerland: Springer, 2022, pp. 432–451.

[16] N. Alnahawi, J. Müller, J. Oupický, and A. Wiesmaier, "SoK: Post-quantum TLS handshake," Cryptol., IACR, USA, Tech. Rep. 2023/1873, 2023. [Online]. Available: https://eprint.iacr.org/2023/1873

[17] 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 *Proc. 20th ACM Int. Conf. Comput. Frontiers*. New York, NY, USA: Association for Computing Machinery, May 2023, pp. 366–374.

[18] R. Gonzalez and T. Wiggers, "KEMTLS vs. post-quantum TLS: Performance on embedded systems," in *Security, Privacy, and Applied Cryptography Engineering* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2022, pp. 99–117.

[19] J. Henrich, A. Heinemann, A. Wiesmaier, and N. Schmitt, *Performance Impact of PQC KEMs on TLS 1.3 Under Varying Network Characteristics* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2023, pp. 267–287.

[20] G. Selander, J. P. Mattsson, and F. Palombini. (Mar. 2024). *Ephemeral Diffie-Hellman Over COSE (EDHOC)*. [Online]. Available: https://www.rfc-editor.org/info/rfc9528

[21] UOSCORE/uEDHOC development team. (2024). *Uoscore/uedhoc*. Accessed: Sep. 19, 2024. [Online]. Available: https://github.com/eriptic/uoscore-uedhoc

[22] Open Quantum Safe Project. (2024). *Liboqs*. Accessed: Sep. 19, 2024. [Online]. Available: https://github.com/open-quantum-safe/liboqs

[23] PQM4 Develop. Team. (2023). *Pqm4*. Accessed: Sep. 19, 2024. [Online]. Available: https://github.com/mupq/pqm4

[24] M. J. Kannwischer, P. Schwabe, D. Stebila, and T. Wiggers, "Improving software quality in cryptography standardization projects," in *Proc. Secur. Standardization Res. (EuroSP Workshops)*, Jun. 2022, pp. 19–30.

[25] M. J. Kannwischer, J. Rijneveld, P. Schwabe, and K. Stoffelen, "PQM4: Testing and benchmarking NIST PQC on ARM cortex-m4," in *Proc. 2nd PQC Standardization Conf., NIST*, 2019. [Online] Available: https://csrc.nist.gov/CSRC/media/Events/Second-PQC-Standardization-Conference/documents/accepted-papers/kannwischer-pqm4.pdf

[26] M. J. Kannwischer, M. Krausz, R. Petri, and S.-Y. Yang, "PQM4: Benchmarking NIST additional post-quantum signature schemes on microcontrollers," Cryptol., IACR, USA, Tech. Rep. 2024/112, 2024. [Online]. Available: https://eprint.iacr.org/2024/112

[27] M.-J. O. Saarinen, "Mobile energy requirements of the upcoming NIST post-quantum cryptography standards," in *Proc. 8th IEEE Int. Conf. Mobile Cloud Comput., Services, Eng. (MobileCloud)*, Aug. 2020, pp. 23–30.

[28] M. Vidaković and K. Miličević, "Performance and applicability of post-quantum digital signature algorithms in resource-constrained environments," *Algorithms*, vol. 16, no. 11, p. 518, Nov. 2023. [Online]. Available: https://www.mdpi.com/1999-4893/16/11/518

[29] B. Halak, T. Gibson, M. Henley, C.-B. Botea, B. Heath, and S. Khan, "Evaluation of performance, energy, and computation costs of quantum-attack resilient encryption algorithms for embedded devices," *IEEE Access*, vol. 12, pp. 8791–8805, 2024.

[30] J. Samandari and C. Gritti, "Post-quantum authentication in the MQTT protocol," *J. Cybersecurity Privacy*, vol. 3, no. 3, pp. 416–434, Jul. 2023. [Online]. Available: https://www.mdpi.com/2624-800X/3/3/21

[31] L. Malina, P. Dobias, P. Dzurenda, and G. Srivastava, "Quantum-resistant and secure MQTT communication," in *Proc. 19th Int. Conf. Availability, Rel. Secur.* New York, NY, USA: ACM, Jul. 2024, pp. 1–8.

[32] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis, "Assessing the overhead of post-quantum cryptography in TLS 1.3 and SSH," in *Proc. 16th Int. Conf. Emerg. Netw. EXperiments Technol.* New York, NY, USA: ACM, Nov. 2020, pp. 149–156, doi: 10.1145/3386367.3431305.

[33] M. Kempf, N. Gauder, B. Jaeger, J. Zirngibl, and G. Carle, "A quantum of QUIC: Dissecting cryptography with post-quantum insights," 2024, *arXiv:2405.09264*.

[34] S. Hristozov, M. Huber, L. Xu, J. Fietz, M. Liess, and G. Sigl, "The cost of OSCORE and EDHOC for constrained devices," in *Proc. 11th ACM Conf. Data Appl. Secur. Privacy*, Apr. 2021, pp. 245–250.

[35] L. Pocero Fraile, A. Fournaris, and C. Koulamas, "Design and performance evaluation of an embedded EDHOC module," in *Proc. 10th Medit. Conf. Embedded Comput. (MECO)*, Jun. 2021, pp. 1–6.

[36] R. Höglund, M. Tiloca, G. Selander, J. P. Mattsson, M. Vučinić, and T. Watteyne, "Secure communication for the IoT: EDHOC and (group) OSCORE protocols," *IEEE Access*, vol. 12, pp. 49865–49877, 2024.

[37] G. Fedrecheski, M. Vučinić, and T. Watteyne, "Performance comparison of EDHOC and DTLS 1.3 in Internet-of-Things environments," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Dubai, UAE, Apr. 2024, pp. 1–6, doi: 10.1109/WCNC57260.2024.10570830.

[38] S. Pérez, J. L. Hernández-Ramos, S. Raza, and A. Skarmeta, "Application layer key establishment for end-to-end security in IoT," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2117–2128, Mar. 2020.

[39] G. Selander, J. P. Mattsson, F. Palombini, and L. Seitz, *Object Security for Constrained RESTful Environments (OSCORE)*, document RFC 8613, Jul. 2019. [Online]. Available: https://www.rfc-editor.org/info/rfc8613

[40] H. Krawczyk, "SIGMA: The 'sign-and-mac' approach to authenticated Diffie–Hellman and its use in the IKE protocols," in *Advances in Cryptology—CRYPTO*, D. Boneh, Ed., Berlin, Germany: Springer, 2003, pp. 400–425.

[41] T. Perrin. (Jul. 2018). *The Noise Protocol Framework*. [Online]. Available: https://noiseprotocol.org/noise.html

[42] E. Barker, L. Chen, A. Roginsky, A. Vassilev, and R. Davis, "Recommendation for pair-wise key-establishment schemes using discrete logarithm cryptography," Comput. Secur. Division, Inf. Technol. Lab., Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST Special Publication 800-56A Revision 3, 2018, doi: 10.6028/NIST.SP.800-56Ar3.

[43] H. Krawczyk and P. Eronen, *CHMAC-based Extract-and-expand Key Derivation Function (HKDF)*, document RFC 5869, May 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5869

[44] C. Bormann and P. E. Hoffman, *Concise Binary Object Representation (CBOR)*, document RFC 7049, Oct. 2013. [Online]. Available: https://www.rfc-editor.org/info/rfc7049

[45] J. Schaad, *Cbor Object Signing and Encryption (COSE)*, document RFC 8152, Jul. 2017. [Online]. Available: https://tools.ietf.org/html/rfc7252

[46] Z. Shelby, K. Hartke, and C. Bormann, *The Constrained Application Protocol (COAP)*, document RFC 7252, Jun. 2014. [Online]. Available: https://tools.ietf.org/html/rfc7252

[47] C. Bormann, *Block-Wise Transfers in the Constrained Application Protocol (COAP)*, document RFC 7959, Aug. 2016. [Online]. Available: https://datatracker.ietf.org/doc/rfc7959/

[48] H. Krawczyk and P. Eronen, *HMAC-Based Extract-and-Expand Key Derivation Function (HKDF)*, document RFC 5869, May 2010. [Online]. Available: https://www.rfc-editor.org/info/rfc5869

[49] J. Schaad, *CBOR Object Signing and Encryption (COSE): Header Parameters for Carrying and Referencing X.509 Certificates*, document RFC 9360, Feb. 2023. [Online]. Available: https://www.rfc-editor.org/info/rfc9360

[50] N. Aragon. (Oct. 2022). *BIKE: Bit Flipping Key Encapsulation*. [Online]. Available: https://bikesuite.org/

[51] D. J. Bernstein. (Oct. 2022). *Classic McEliece*. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions

[52] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. D. Feo, B. Hess, A. Jalali, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, J. Renes, V. Soukharev, D. Urbanik, G. Pereira, K. Karabina, and A. Hutchinson. *Supersingular Isogeny Key Encapsulation*. Accessed: Sep. 19, 2024. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions

[53] W. Castryck and T. Decru, "An efficient key recovery attack on SIDH," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), C. Hazay and M. Stam, Eds., Cham, Switzerland: Springer, 2023, pp. 423–447.

[54] D. Robert, "Breaking SIDH in polynomial time," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), C. Hazay and M. Stam, Eds., Cham, Switzerland: Springer, 2023, pp. 472–503.

[55] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. (2017). *Falcon: Fast-Fourier Lattice-based CompactSignatures Over NTRU*. [Online]. Available: https://falcon-sign.info/falcon.pdf

[56] (2023). *Post-quantum Signatures Zoo*. Accessed: Oct. 29, 2024. [Online]. Available: https://pqshield.github.io/nist-sigs-zoo/

[57] L. Ducas, E. W. Postlethwaite, L. N. Pulles, and W. V. Woerden, "Hawk: Module LIP makes lattice signatures fast, compact and simple," in *Advances in Cryptology—ASIACRYPT* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2022, pp. 65–94.

[58] J. H. Cheon, H. Choe, J. Devevey, T. Güneysu, D. Hong, M. Krausz, G. Land, M. Möller, D. Stehlé, and M. Yi, "HAETAE: Shorter lattice-based fiat-shamir signatures," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2024, no. 3, pp. 25–75, Jul. 2024. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/11669

[59] (Mar. 2025). *Korean PQC Competition Round 2*. [Online]. Available: https://www.kpqc.or.kr/competition_02.html

[60] IANA. (2024). *Cbor Object Signing and Encryption (COSE)*. [Online]. Available: https://www.iana.org/assignments/cose/cose.xml

[61] (2024). *Ephemeral Diffie–Hellman Over COSE (EDHOC)*. [Online]. Available: https://www.iana.org/assignments/edhoc/edhoc.xhtml#edhoc-cipher-suites

[62] Libcoap Develop. Team. (2024). *Libcoap: A C-implementation of Coap—Constrained Application Protocol*. Accessed: 2024-09-30. [Online]. Available: https://libcoap.net/

[63] (2023). *MUPQ: The Minimalist Post-Quantum Cryptography Library*. [Online]. Available: https://github.com/mupq/mupq

[64] (2018). *Post-Quantum Cryptography FAQs*. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/faqs

[65] NIST. (2024). *FIPS 204 (Initial Public Draft): Stateless Hash-Based Digital Signature Standard*. [Online]. Available: https://csrc.nist.gov/pubs/fips/204/ipd

[66] (2024). *PQM4 Benchmarks With NIST Initial Drafts for FIPS 203, 204*. [Online]. Available: https://github.com/mupq/pqm4/blob/8d44b724396ddbc0db55d5de93bec252cedb9c04/benchmarks.md

[67] NIST. *PQM4 Benchmarks With NIST Final Standards for FIPS 203,204*. Accessed: 29-10-2024. [Online]. Available: https://github.com/mupq/pqm4/blob/79a0ddf3399067eb2de3b14d92331411b195bf10/benchmarks.md

[68] D. Moody. *NIST Urge To Focus on Cortex-M4*. Accessed: Oct. 29, 2024. [Online]. Available: https://csrc.nist.gov/CSRC/media/Presentations/the-2nd-round-of-the-nist-pqc-standardization-proc/images-media/moody-opening-remarks.pdf

[69] (Sep. 2024). *Zephyr Project: Scalable Real-time Operating System (RTOS) for Embedded Devices*. [Online]. Available: https://zephyrproject.org/

[70] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. "Crystals-dilithium: A lattice-based digital signature scheme," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2018, no. 1, pp. 238–268, Feb. 2018.

[71] I. Corporation. (Sep. 2024). *Tinycrypt Cryptographic Library*. [Online]. Available: https://github.com/intel/tinycrypt/blob/master/documentation/tinycrypt.rst

[72] D. Beer. (Sep. 2024). *Curve25519 and Ed25519 for Low-Memory Systems*. [Online]. Available: https://www.dlbeer.co.nz/oss/c25519.html

[73] N. Semiconductor. (Sep. 2024). *Nrf52840 Product Specification*. [Online]. Available: https://docs.nordicsemi.com/bundle/ps_nrf52840/page/keyfeatures_html5.html

[74] PICO Technol. (2024). *Picoscope 5000 Series Datasheet*. Accessed: Sep. 19, 2024. [Online]. Available: https://www.picotech.com/download/datasheets/MM040.en-8.pdf

LIDIA POCERO FRAILE received the Dipl.Eng. degree in telecommunication engineering from the Higher Technical School of Telecommunications Engineering, University of Valladolid, Spain, and the M.Sc. degree in hardware and software integrated systems (HSIS) from the Department of Electrical and Computer Engineering, University of Patras, Greece. She is currently pursuing the Ph.D. degree with the Department of Informatics and Telecommunications, University of Ioannina, specializing in "Efficient and Secure Industrial IoT (IIoT) Components and Systems for Smart and Safe Environments."

She is a Collaborating Researcher with the Industrial Systems Institute, R. C. Athena, Greece. She has extensive experience in building automation, WSNs, and lightweight security protocols. Since 2012, she has contributed to numerous European and Greek research and development projects. Her research interests include the IoT security for industrial and critical infrastructures, including secure communication protocols for embedded systems and the transition to post-quantum cryptography.

GEORGIOS TASOPOULOS received the Electrical Engineering and Computer Technology Diploma (integrated B.Sc. and M.Sc.) degree from the University of Patras, Greece, in 2021. He is currently pursuing the Ph.D. degree with the Informatics Institute, University of Amsterdam, The Netherlands. From 2021 to 2023, he was a Collaborating Researcher with the Industrial Systems Institute/Athena Research Center, Greece. His research interests include post-quantum cryptography, embedded systems security, and homomorphic encryption for privacy-preserving applications.

CHRISTOS KOULAMAS (Senior Member, IEEE) received the Diploma degree in computer engineering and informatics and the Ph.D. degree in electrical and computer engineering from the University of Patras, Greece, in 1994 and 2004, respectively.

He is currently a Research Director with the Industrial Systems Institute (ISI), R. C. Athena, Greece. He has more than 25 years of experience in research and development, in the areas of real-time distributed embedded systems, industrial networks, wireless sensor networks, and their applications in various industry sectors. He is the author or co-author of more than 80 scientific publications in international journals, conferences, and books, and of more than 110 publicly available or confidential technical reports. His current research interests include cyber-physical systems (CPS), the Industrial Internet of Things (IIoT) technologies, and their applications in smart environments.

Dr. Koulamas has served as an Editorial Board Member of *Sensors* journal, as a PC/TPC member of many IEEE and other conferences and workshops, and as a member of the Board of European Research Consortium for Informatics and Mathematics (ERCIM). He has also served as a Guest Editor for *Electronics* and *Sensors* journals (MDPI)

RAYMOND K. ZHAO received the B.Eng. degree in computer science and technology from Zhejiang University, China, in 2015, the master's degree in network and security from Monash University, Australia, in 2017, and the Ph.D. degree from the Faculty of Information Technology (FIT), Monash University, in 2022. Since November 2022, he has been a Postdoctoral Fellow with CSIRO's Data61. His main research interests include efficient and secure implementation techniques for post-quantum cryptographic applications and protocols.

NAZATUL HAQUE SULTAN received the Ph.D. degree from Indian Institute of Information Technology Guwahati. He is currently a Research Scientist with CSIRO's Data61. His research interests include privacy-enhancing technologies, applied cryptography, AI/ML security, and post-quantum cryptography (PQC).

FRANCESCO REGAZZONI (Member, IEEE) received the Master of Science degree from Politecnico di Milano and the Ph.D. degree from Università della Svizzera italiana. He held research positions with Université Catholique de Louvain and the Technical University of Delft and has been a Visiting Researcher at several institutions, including NEC Laboratories America, Ruhr University of Bochum, and EPFL Lausanne. He is currently an Associate Professor with the University of Amsterdam and he is also affiliated with Università della Svizzera italiana. His main research interests include the security of IoT devices and embedded systems, covering in particular design automation for security, physical attacks and countermeasures, post-quantum cryptography, and efficient implementation of cryptographic primitives.

APOSTOLOS P. FOURNARIS (Member, IEEE) received the Ph.D. degree in public key cryptography systems design from the Electrical and Computer Engineering Department, University of Patras, Patras, Greece, in 2008.

He was with the Sophia Antipolis Hitachi Europe SAS R-D Centre for two years as a Researcher. He is currently a Researcher and the Director of the Industrial Systems Institute, R. C. Athena, Greece. He was also a Visiting Sessional Lecturer with Monash University, Melbourne, VIC, Australia, where he lectured courses on cryptography, computer, and network security. He is the author of more than 80 research articles. His research interests include asymmetric cryptography, side channel attacks and analysis, hardware attack resistance, security–privacy, and trust in IoT systems.

Dr. Fournaris is a member of the International Association for Cryptologic Research, the IEEE Computer Society, and the IEEE Circuits and System Society.