Towards Quantum Resilient IoT: A Backward-Compatible Approach to Secure BLE Key Exchange Against Quantum Threats

Tao Liu
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Australia
t24.liu@hdr.qut.edu.au

Gowri Ramachandran
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Australia
g.ramachandran@qut.edu.au

Raja Jurdak
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Australia
r.jurdak@qut.edu.au

***Abstract*—There's a significant move towards the adoption of Post-Quantum Cryptography (PQC). While there have been initiatives to transition conventional TCP/IP-based networks to PQC, Quantum Resilient Internet of Things (IoT) networks have not been as widely discussed. Presently, Bluetooth Low Energy (BLE) employs the Elliptic Curve Diffie-Hellman algorithm for Secure Connection (SC) pairing, which is vulnerable to quantum threats. In this study, we introduce a backward-compatible Post-Quantum Key Exchange (PQKE) protocol for BLE, utilizing the Kyber-512 algorithm that has been adopted by the National Institute of Standards and Technology as a post quantum Key Encapsulation Mechanism. Although Kyber-512 is quantum resilient, it has large key pairs and ciphertexts, which presents critical challenges for the limited computational resources of IoT devices. Our performance assessment reveals that with an Attribute Protocol Maximum Transmission Unit (ATT MTU) of 65 bytes, the pairing time increases by approximately 9 folds using our PQKE in comparison to the traditional BLE SC pairing, mainly due to increased data transmission. Nevertheless, by employing a larger ATT MTU, the pairing time of our PQKE mechanism can be minimized to be of the same order of magnitude as current pre-quantum key exchange for BLE. We therefore advocate for the adoption of larger ATT MTU sizes in quantum resilient BLE pairing to ensure the performance and usability of the technology in a post-quantum world.**

***Index Terms*—Post-Quantum Cryptography, Internet of Things, Bluetooth Low Energy**

I. Introduction

Due to recent development in quantum computing, the emergence of scalable quantum computers is being anticipated. A quantum computer is a system that leverages principles of quantum mechanics to compute with quantum bits (or qubits) instead of digital bits, allowing it to execute quantum algorithms, which can solve certain mathematical problems with great efficiency. An important quantum algorithm critical to modern Public-Key Cryptography (PKC) is Shor's period-finding algorithm, which can be utilized to solve the integer factorization and the discrete logarithms problems in polynomial time [25]. Both problems are special cases of the Hidden Subgroup Problem (HSP) over finite Abelian groups, and they form the basis and security guarantee of modern PKC. If the HSP problems truly can be solved efficiently by quantum computers, then the landscape of data encryption and cybersecurity in general would be serious disrupted.

Fortunately, there are still years until a practical quantum computer capable of threatening cryptography - or so-called Cryptographic-Relevant Quantum Computer (CRQC) - becomes available. Today's prototype quantum computers exist as Noisy Intermediate-Scale Quantum (NISQ) systems with limited number of qubits (and processing power). At the current stage, NISQ systems have difficulties scaling up due to the challenge of quantum error correction [22]. It is anticipated, however, that the invention of a fault-tolerant quantum computer would eventually endanger all existing PKC schemes. In a recent survey, experts suggest that a CRQC capable of breaking 2048-bit Rivest-Shamir-Adleman (RSA) cipher with Shor's algorithm within 24 hours is likely to be built within the next 15-30 years [17].

As such, it is important for the Internet to migrate to Post-Quantum Cryptography (PQC), which cannot be solved efficiently even by quantum computers. Currently, projects for PQC migration are being spearheaded by global institutions, such as the US National Institute of Standards of Technology (NIST)[18]. PQC migration is an urgent matter even though we are still years away from fault-tolerant CRQCs, and the reasons are two-fold: (1) existing infrastructures (i.e. common protocols, embedded software libraries, and hardware crypto-accelerators) need time and resource to upgrade; (2) the potential for a "Save-Now, Decrypt-Later" (SNDL) attack exists. In this scenario, adversaries could hoard encrypted data, only to retrospectively decrypt it once CRQCs become operational [12]. It is noteworthy that the PQC migration also faces challenges, and one significant challenge is that PQC algorithms are generally more complex, with significantly larger key and data sizes which may cause issues with computation and transmission costs [4].

To date, ongoing research mostly focus on PQC migration at broad perspective, prioritizing quantum resistance for general-purpose networks. Quantum resistance for Internet-of-Things (IoT) architectures, on the other hand, is a topic that received less attention. However, as IoT is an emerging technology and

979-8-3503-7025-6/24/$31.00 ©2024 IEEE
DOI 10.1109/IoTDI61053.2024.00019

its popularity is on the rise, quantum resilience for IoT is also of great importance [23]. In essence, an IoT network comprises a vast array of interconnected, intelligent devices capable of autonomously sharing data and responding to environmental changes[15]. To fulfill this purpose, an IoT network often utilizes a large number of heterogeneous, lightweight, and resource-constrained sensors and actuators, and such hardware may not have the capability to compute PQC algorithms in a safe and timely manner [14]. According to [21], the challenges of PQC transition include (1) impact on network performance; (2) less-proven security strength and (3) vulnerabilities related to software implementation. For IoT systems, performance issues may be particularly pronounced due to the extra constrains on device resources.

In this paper, we investigate the performance feasibility of integrating quantum resilience for the Bluetooth Low Energy (BLE) stack, which is a market-leading RF technology commonly used by IoT devices. According to Bluetooth Core Specification v5.4 [26], BLE uses the Elliptic Curve Diffie-Hellman (ECDH) algorithm, which offers no quantum resilience, to pair two devices with no prior knowledge of each other. To demonstrate that BLE can be made quantum resilient, we propose a Post-Quantum Key Exchange (PQKE) mechanism over BLE's Generic Attribute Profile (GATT) with CRYSTALS-Kyber, a quantum resilient Key Encapsulation Mechanism (KEM) selected by NIST for standardization. Furthermore, we analyze the performance of our implementations in terms of temporal cost (i.e. pairing latency), which includes both cryptographic computation time and RF transmission time. Our proposed contributions include:

1) we present a novel yet backward-compatible PQKE which retrospectively supports older BLE-enabled devices.
2) we show that quantum resilience is achievable and feasible for the BLE technology.
3) we evaluate and analyze the performance of PQKE. Our evaluation can be used for reference by future design of quantum resilient IoT.

The rest of this paper is structured as follows: in section II, we discuss existing works that are relevant to the issue of PQC migration for IoT. In section III, we present preliminary knowledge, which includes a detailed quantum threat model, as well as the tools we use to address the threat. The experimental methodology is discussed in section IV, and the results are presented and discussed in section V. Finally, we summarize our findings in section VI and put forward recommendations.

II. Related Work

The call for migration toward PQC has gained traction in the latest years. Currently, the most prominent project is being spearheaded by NIST, which is a process initiated in 2017 to "solicit, evaluate, and standardize one or more quantum resilient public-key cryptographic algorithms" [20]. In a most recent update, three algorithms, including one KEM and two digital signature schemes were standardized, as listed in Table I. Furthermore, drafts of Federal Information Processing Standard (FIPS) were published for these algorithms [19]. It is anticipated that these algorithms are on track to become globally recognized and adapted.

Algorithm	**NIST Standardized Name**	**Feasibility for IoT**
CRYSTALS-Kyber	Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM)	Feasible
CRYSTALS-Dilithium	Module-Lattice-Based Digital Signature (ML-DSA)	Manageable
SPHINCS+	Stateless Hash-Based Digital Signature (SLH-DSA)	Infeasible

TABLE I: List of PQC algorithms currently being standardized by NIST.

To date, existing literature has indicated the potential of lightweight IoT nodes to compute NIST-selected PQC algorithms with satisfactory performance. This consensus laid the groundwork for our preliminary speculation in the feasibility of a quantum resilient BLE pairing mechanism. Perhaps the most compelling evidence supporting this notion is the PQM4 project [13]. PQM4 presents a comprehensive implementation of NIST-approved PQC algorithms, and these implementations are optimized for the ARM Cortex-M4 processor. Given that a multitude of RF communication System-on-Chips (SoC) are based on the M4 architecture, the implications and significance of PQM4's contributions cannot be understated.

Another significant focus within PQC research has been on the Post-Quantum migration of the Transport Layer Security (TLS) protocol. Given that the TLS protocol is widely adopted as the go-to standard for securing client-server online communication [30], effectively implementing PQTLS could infuse quantum resistance into several prevalent web protocols that utilize TLS, such as HTTP, FTP, SMTP, and IMAP. It's logical to assume that, since some IoT network protocols, like MQ Telemetry Transport (MQTT), also leverage TLS, transitioning to Post-Quantum TLS, or PQTLS, would also benefit IoT security. Research examining the performance of PQTLS in IoT contexts includes [7], [24], and [9], all of which suggest that lattice-based PQC algorithms are somewhat feasible IoT. Particularity, existing work suggest that the performance of Kyber512/ML-KEM-512 is on par with ECDH and even better than RSA2048 in terms of computation time, giving us hope on the possibility of quantum resilient IoT. The study by [10] recognizes the computational and transmission challenges posed by large Post-Quantum signatures for IoT. As a solution, the study proposes a TLS-like protocol that employs KEMs for both key exchange and authentication, and demonstrates that such an approach is more time- and energy-efficient.

Nevertheless, solutions grounded in TLS cater predominantly to the "Internet" component of IoT since TLS requires TCP/IP to operate [30]. This underscores a limitation: PQTLS on its own is not a comprehensive solution for quantum resilient IoT. A substantial portion of IoT nodes communicate via RF technologies like Bluetooth and Zigbee [27], and these don't operate on TLS. Instead, they have built-in security protocols. For instance, Bluetooth Low Energy (BLE) uses

Bluetooth Security Manager Protocol (SMP) and relies on ECDH for device pairing and mesh network key provisioning. Similarly, Zigbee's most recent iteration (R23) also employs ECDH for pairing. Given ECDH's vulnerability to quantum threats, there's a clear need to revisit and possibly revamp the pairing mechanisms of RF protocols.

Regarding Post-Quantum security for RF-based communication, the only existing publication we identified is [6]. This study delineates an interesting setup wherein PQC is integrated into BLE. To achieve this, the authors implement an IPv4 network over BLE using customized software, which they subsequently secured with PQTLS. While this methodology is operational, its practicality is questionable. Specifically, the size of code for implementing IP over BLE, combined with the PQTLS library is substantial, casting doubts on the genuine lightweight nature of this assemblage.

Synthesizing the information from the existing body of research, it becomes evident that the current academic discourse lacks an efficient scheme for imbuing RF-based communication protocols with quantum resistance. Addressing this issue, our research introduces a novel PQKE mechanism for BLE, which is a representative RF technology with high popularity within the IoT landscape.

III. Preliminaries

In this section, we discuss the preliminary knowledge, which form the foundation of our justification for the necessity and usefulness of the proposed PQKE.

A. The "Save-Now, Decrypt-Later Attack

Shown in Figure 1 is the model of a quantum SNDL attack, against which our proposed PQKE mechanism is meant to defend. Essentially, this threat model is a passive eavesdropping attack enabled by a CRQC. In a normal scenario, two IoT devices can establish a shared Key ("pairing") with each other using a certain PKC algorithm, and continue to communicate over symmetric encryption using the shared key. For BLE SC pairing, the PKC algorithm used is ECDH P-256 and the symmetric cipher is AES-128-CCM. [26].

The ultimate assumption behind the quantum threat is that a sophisticated CRQC is capable of breaking PKC. Furthermore, as an SNDL attack, a quantum threat actor can surveillance the RF packets using specialized hardware such RF packet sniffers and store the data. Then, when a CRQC becomes available, the threat actor may use it to decrypt the key establishment process and recover the shared key, and proceed to exploit subsequent messages. In this scenario, the confidentiality and integrity of data is completely broken. However, we note that this threat model has two limitations:

1) **The attacker must be within the proximity**. As RF packets are not relayed through the Internet, a remote threat actor is not likely to access the transmission.
2) **The attacker must be present during the key establishment**. If the RF packets used for key establishment were not captured and the attacker can only access the subsequent encrypted packets. this attack would not be viable.[1]

Despite its limitations, the SNDL attack has the theoretical capability of revealing critical secrets transmitted in the present at a later time, thus a defence mechanism should be available in a timely manner.

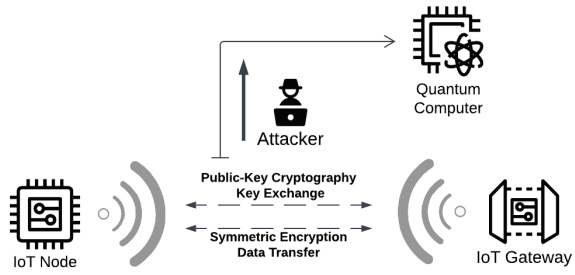

Fig. 1: Threat model considered in this work: passive eavesdropping on encrypted RF communication enabled by QC.

B. CRYSTALS-Kyber

For establishing a secret under the threat of a quantum adversary described above, the CRYSTALS-Kyber KEM [5] is a suitable PQC algorithm for its recognized security [2] and efficiency on lightweight platforms [24]. Essentially, Kyber is based on the hardness of solving the Learning-With-Errors (LWE) problem over module lattices and has strong ciphertext indistinguishably. The Kyber family offers three parameter sets, dubbed Kyber-512, Kyber-768 and Kyber-1024, with increasing security strength and computation cost. A comparison of security strength between Kyber and ECDH P-256, which is the algorithm used by BLE Secure Connection (SC) Pairing, is presented in Table II. As shown in the table, key establishment over Kyber grants BLE pairing quantum resilience, but at the cost of significantly increased key sizes and ciphertext transmission.

	PQ Security Strength[2]	**Classic Security Strength**	**PK Size (Bytes)**	**SK Size (Bytes)**	**c Size (Bytes)**
ECDH p-256	No Security	128-bit	32	32	-
Kyber-512	Category 1	128-bit	800	1632	768
Kyber-768	Category 3	192-bit	1184	2400	1088
Kyber-1024	Category 5	256-bit	1568	3168	1568

TABLE II: Comparison of key sizes, ciphertext sizes and security strength between the Kyber family and ECDH P-256.

For our purpose, we choose the most lightweight version, Kyber-512. This algorithm maintains the 128-bit classic security strength adapted throughout Bluetooth Specification while

[1]There is a "quantum search" algorithm which in theory can speedup exhaustive key-searching [11], but such topic is beyond our scope.

offering quantum resilience at NIST PQC Security Category 1. Should the need arise, our methodology can easily be expanded and applied to other parameter sets of Kyber, or other Post-Quantum KEMs.

Essentially, Kyber-512 has three main cryptographic functions:

- `Kyber512.KeyGen()` accepts no input, requires randomness and outputs a Kyber-512 key pair: 800-byte public key PK and 1632-byte secret key SK. while PK is to be made public, SK must be kept secret by the device generated it.
- `Kyber512.Encap()` accepts PK, a 32-byte random secret z to be shared, and outputs a 768-byte ciphertext c.
- `Kyber512.Decap()` accepts SK, ciphertext c and outputs the shared secret z.

As recommended by NIST [19], the shared secret z can be directly used as symmetric encryption key, or used to derive keys. Thus, for our propose, we consider the key exchange complete once z exists in the memory of both devices. In practice, further securing the key from being extracted via conventional attacks may also be necessary.

C. Bluetooth Low Energy

Bluetooth Low Energy (BLE) is a power-efficient variant of the classic Bluetooth technology, specifically designed to support short bursts of data transmission while consuming minimal energy [29]. In this section, we discuss key features of BLE which is crucial to the optimization of the proposed PQKE.

1) BLE Connection and Architecture: A BLE connection follows the client-server paradigm, where a Peripheral acts as the server and responds to the requests made by a Central. The BLE stack architecture consists of three components: the Host, the Controller, and the Host Controller Interface (HCI), as shown in Figure 2. The Host constitutes the upper layer of the BLE stack and envelops the higher-level protocols such as GATT and SMP. Below the Host lies the Controller, which includes the Link Layer (LL) and the Physical Layer (PHY). Bridging the gap between the Host and the Controller is the HCI, which is responsible for transporting commands, events, and data between the Host and the Controller.

2) Data Structure and Access: Within the BLE Host, ATT defines how data is structured and accessed on a BLE device, and GATT builds upon ATT to introduce the concept of profiles, services, and characteristics. Essentially, GATT organizes data into a hierarchical structure, making them more usable for applications. The top of the GATT hierarchy is a GATT profile, the functionalities of which is by the services it includes. A GATT services is a collection of GATT characteristics, which are actual pieces of data exposed on the peripheral. A characteristic can be at most 512 bytes long, and can be read and written by the central device with correct permissions. The peripheral can also initiate transmission using Notify and Indication methods.

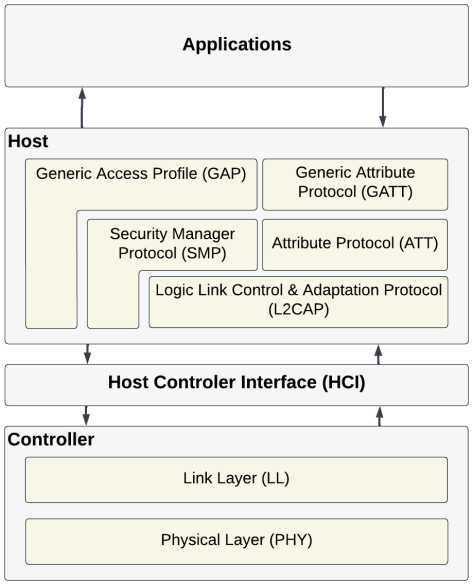

Fig. 2: Architecture of the BLE protocol stack.

3) Data Segmentation: The size of ATT Maximum Transmission Unit (MTU) is crucial to BLE throughput, and therefore the performance of our proposed mechanism, as longer GATT characteristics must be transmitted through multiple ATT packets. Theoretically, using larger MTU size can improve transmission throughput, albeit at the cost of larger memory footprint and risk of data corruption [3]. Additionally, some platforms - such as the iOS - only allows smaller MTUs, further complicating the performance issue of quantum resilient pairing. On the L2CAP layer, ATT packets are further fragmented into L2CAP fragments, which originally had a maximum Protocol Data Unit (PDU) of 27 bytes. However, with the Data Length Extension (DLE) feature introduced in Bluetooth 4.2, the size of L2CAP packets can be extended to at most 251 bytes, increasing the theoretical throughput by 250% [28].

4) PQC Transition Considerations: For the security of BLE, the SMP manages all functions related to device pairing, key distribution, and data encryption. This includes the SC pairing methods introduced in Bluetooth 4.2, which employ ECDH to establish a shared secret between devices. Given the evolving landscape, it's intuitive that a quantum resilient pairing mechanism be developed within SMP.

However, the transition to PQC presents its own set of challenges, a primary one being backward compatibility. For perspective, consider the shift from BLE's Legacy Pairing to SC: while SC has become the best practice for device pairing, Legacy Pairing remains unbroken and usable for older devices, albeit with weaker security. This analogy, unfortunately, doesn't apply to PQC migration, as quantum readiness is of a binary nature: a cryptographic algorithm is either quantum resilient or not. As such, should a future BLE update

introduce quantum resilient pairing, devices adhering to earlier specifications could become obsolete for not supporting the new mechanism.

For this problem, a potential solution exists: if quantum resilient pairing was executed through a GATT service, legacy hardware could be updated to encompass this new service within their existing GATT profiles, thereby preserving their relevancy and utility. Furthermore, as PQC standardization is still in its nascent stage, prematurely integrating a novel cryptosystem into the core specification could be imprudent. Such an action might necessitate subsequent modifications should the cryptosystem be deemed non-optimal or unsafe. In contrast, updates to a self-contained GATT service specification would represent a more modular approach, mitigating potential complexities associated with specification overhauls yet still provide security.

IV. Methodology

In this section, we discuss the research methodology. First, we discuss the architecture of our proposed PQKE. Then, we delve into the matter of performance optimization and discuss the setup involving ATT MTU and L2CAP PDU parameters. Furthermore, we lay down information about specific hardware we use for experiment, and justify the choices.

A. Overview of Proposed PQKE

In an overview, our PQKE service leverages three GATT characteristics, `Ctrl_point`, `PK_out` and `c_in`. Their purposes and allowed GATT access permissions are shown in Table III. A sequence diagram of our proposed PQKE mechanism is presented in Figure 3. Essentially, there are six key steps in our proposed mechanism:

1) **Initialization.** The central sends a request to the peripheral and requests key exchange.
2) **Key Generation.** The peripheral uses `Kyber512.KeyGen()` to generate a PK and SK.
3) **Public Key Transmission.** The peripheral transmits PK to the central via BLE. As the length of PK is greater than the maximum characteristic size allowed (512 bytes), PK is sent via multiple fragments using peripheral-initiated transmission (i.e. *GATT Indicate*).
4) **Secret Encapsulation.** The central uses `Kyber512.Encap()` to produce z and c.
5) **Ciphertext Transmission.** The central transmits c to the peripheral using GATT Write.
6) **Secret Decapsulation.** The peripheral decapsulates c and obtains z with `Kyber512.Decap()`.

Note that the skeleton implementation of PQKE described in this document offers just the essential key exchange functionalities. Typically, standardized BLE services approved by the Bluetooth SIG contain additional GATT characteristics and sophisticated software logic to support peripheral functionalities, such as error handling and state management. However, for the purpose of evaluating the performance baseline, the skeleton implementation should reasonably suffice.

GATT Character	Purpose	Access Permissions
`Ctrl_point`	Control logic flow with opcodes	Read, Write
`PK_out`	Transmit PK	Indicate
`c_in`	Receive c	Write

TABLE III: GATT Characteristics of the proposed PQKE service implementation.

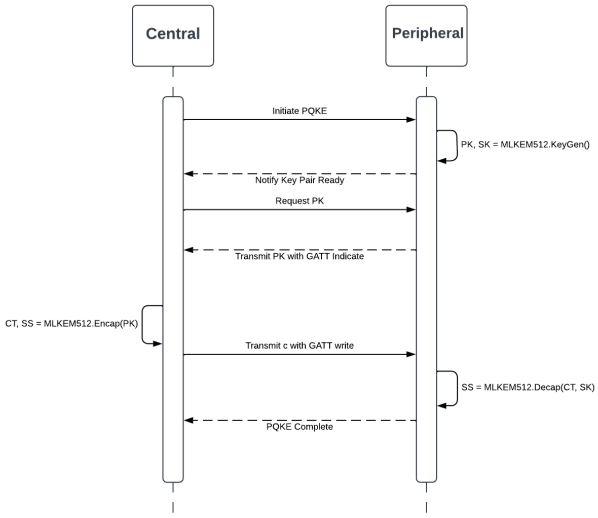

Fig. 3: Sequence Diagram of proposed PQKE using Kyber-512

B. Equipment and Tools

We utilized an Intel NUC11PAHi50Z mini-PC, equipped with integrated Bluetooth 5.2 support, as a BLE Central. Communication with the Peripheral was facilitated through a Python script leveraging Bleak v0.21.1[3].

The chosen Peripheral was a Particle Argon [4] development board, containing an nRF52840 SoC that supports Bluetooth 5.0, along with both 1M and 2M Physical Layers (PHY). The nRF52840 chip, built around the ARM Cortex-M4F, is a predominant microprocessor in the IoT landscape[8]. Additionally, ARM Cortex-M4 is also the main target for PQC performance benchmarking [2, 13]. Given its prominence in real-world IoT implementations, the Particle Argon offers a representative hardware selection for our experiments.

The firmware on the Peripheral is developed over Zephyr RTOS v3.4 [5], which is chosen for several advantages over alternatives [31]: Zephyr RTOS is open-source backed by the Linux Foundation, and features built-in BLE connectivity and high modularity, which means its footprint is minimized by only compiling libraries in use via its Kconfig system. With support for hundreds of boards over multiple architectures such as ARM Cortex M, Intel x86, ARC and RISC-V, Zephyr

[3]https://pypi.org/project/bleak/
[4]https://docs.particle.io/argon/
[5]https://zephyrproject.org/

has a rapidly growing ecosystem as well as a large support community.

Additionally, we use the Wireshark [6] network protocol analyzer with a BLE sniffer (programmed nRF52840 USB Dongle[7]) to examine RF packets in detail and confirm the correct execution of experimental setups.

C. Connectivity Parameters

The configuration specifics for data segmentation at both ATT and L2CAP layers are detailed in Table IV. The rationale for the selection of each set parameter is as follows:

- 23 is the minimum ATT MTU allowed.
- 65 is the minimum ATT MTU required when SMP is enabled (i.e. used by SC pairing over ECDH P-256).
- 185 is the maximum ATT MTU supported by iOS devices.
- All other ATT MTU values divide the 800-byte PK into optimal-sized segments (of 100, 200 and 400 bytes).

For all configurations, the 2M Bluetooth PHY is used for its superior data rate (albeit at the cost of connection range).

Whilst we assume that increasing ATT MTU would improve throughput and by extension PQKE performance, another essential point to understand is the impact of data segmentation (and the lack thereof) at L2CAP layer. To achieve this, we use the DLE feature and extend the L2CAP PDU to match ATT MTU size, effectively bypassing L2CAP layer segmentation, and observe the performance. However, for this to work, ATT MTU must be increased to relay the same data volume as a single transmission must contain both ATT and L2CAP headers. For example, consider a scenario where the Peripheral intends to Indicate a 100-byte PK fragment to the Central:

- **Without DLE:** The necessary ATT MTU is 104, accounting for the additional 4-byte ATT header. This culminates in a total transmission of 107 bytes, which incorporates a 3-byte L2CAP header. The transmission is split across four L2CAP fragments, each restricted to a maximum of 27 bytes.
- **With DLE:** Both the ATT header and the L2CAP header are encapsulated within an ATT MTU of 107, rendering a L2CAP PDU of the same length. This facilitates transmitting the entirety of the data in a single stride.

Furthermore, when the focus is on a fixed ATT MTU, the data transfer dynamics alter slightly. For instance, an ATT MTU of 185 bytes can ferry 181 bytes of data when L2CAP-layer segmentation isn't a constraint. However, to entirely forego L2CAP-layer segmentation, only 178 bytes of actual data can be incorporated into one singular transmission.

For each combination of ATT MTU and L2CAP PDU, we executed the PQKE at varied device distances, including 0.05m, 0.50m, 1.00m, 1.00m without Line-of-Sight (LoS), and 3.00m. To estimate the average RSSI for connections at these specific distances, we took 100 measurements for each respective distance.

[6]https://www.wireshark.org/

[7]https://www.nordicsemi.com/Products/Development-hardware/nrf52840-dongle

	PK **Indicate size**	c **Prep. Write size**	**ATT MTU**	**L2CAP PDU**	**DLE**
(1)	19	18	23	27	No
(2)	61	60	65	27	No
(3)	100	99	104	27	No
(4)	181	180	185	27	No
(5)	200	199	204	27	No
(6)	400	399	404	27	No
(7)	58	56	65	65	Yes
(8)	100	98	107	107	Yes
(9)	178	176	185	185	Yes
(10)	200	198	207	207	Yes
(11)	400	398	404	251	Yes

TABLE IV: ATT and L2CAP layer Segmentation configurations

Distance	**LoS**	**Average RSSI**	$\overline{T}_{SC}$ **(s)**
0.05m	Yes	-44.74	0.3286
0.5m	Yes	-52.84	0.3464
1m	Yes	-62.63	0.3598
1m	No	-67.24	0.3823
3m	Yes	-65.30	0.4702

TABLE V: List of configurations for the distance between Central and Peripheral. $\overline{T}_{SC}$ represents the time of Secure Connection "Just Works" pairing.

D. Performance Baseline

To quantify the performance overhead of our proposed PQKE against traditional (pre-quantum) BLE pairing, we gathered data on the pairing time period between the Central and the Peripheral using Secure Connection (SC) "Just Works" (JW) pairing, which is essentially a key exchange without user interference (thus no Man-in-the-Middle protection) using ECDH P-256.

Zephyr RTOS employs the TinyCrypt cryptographic library [8] for its implementation of ECDH. Instead of injecting timing functions into the lower-level implementation, Wireshark and the BLE sniffer can be used to estimate the pairing time as transmissions are traced with specific timestamps. Our specific measurement revolves around the time span bridging two pivotal events: the initiation of the pairing procedure marked by the Pairing Request (SMP opcode `0x01`) and the validation of the key establishment as signified by the SMP Pairing DHkey Check (SMP opcode `0x0d`).

For every distance setup outlined in Table V, the SC pairing process is replicated 10 times. The resultant average timing serves as a benchmark against the time cost of our proposed PQKE is then contrasted.

V. Results and Discussion

This section examines the influence of ATT and L2CAP layer segmentation on the time efficiency of PQKE. Table VI presents the average PQKE time ($\overline{T}_{pqke}$) along with its standard deviation (SD). For a clearer visual representation of this data, we have provided violin plots in Figure 4.

[8]https://github.com/intel/tinycrypt

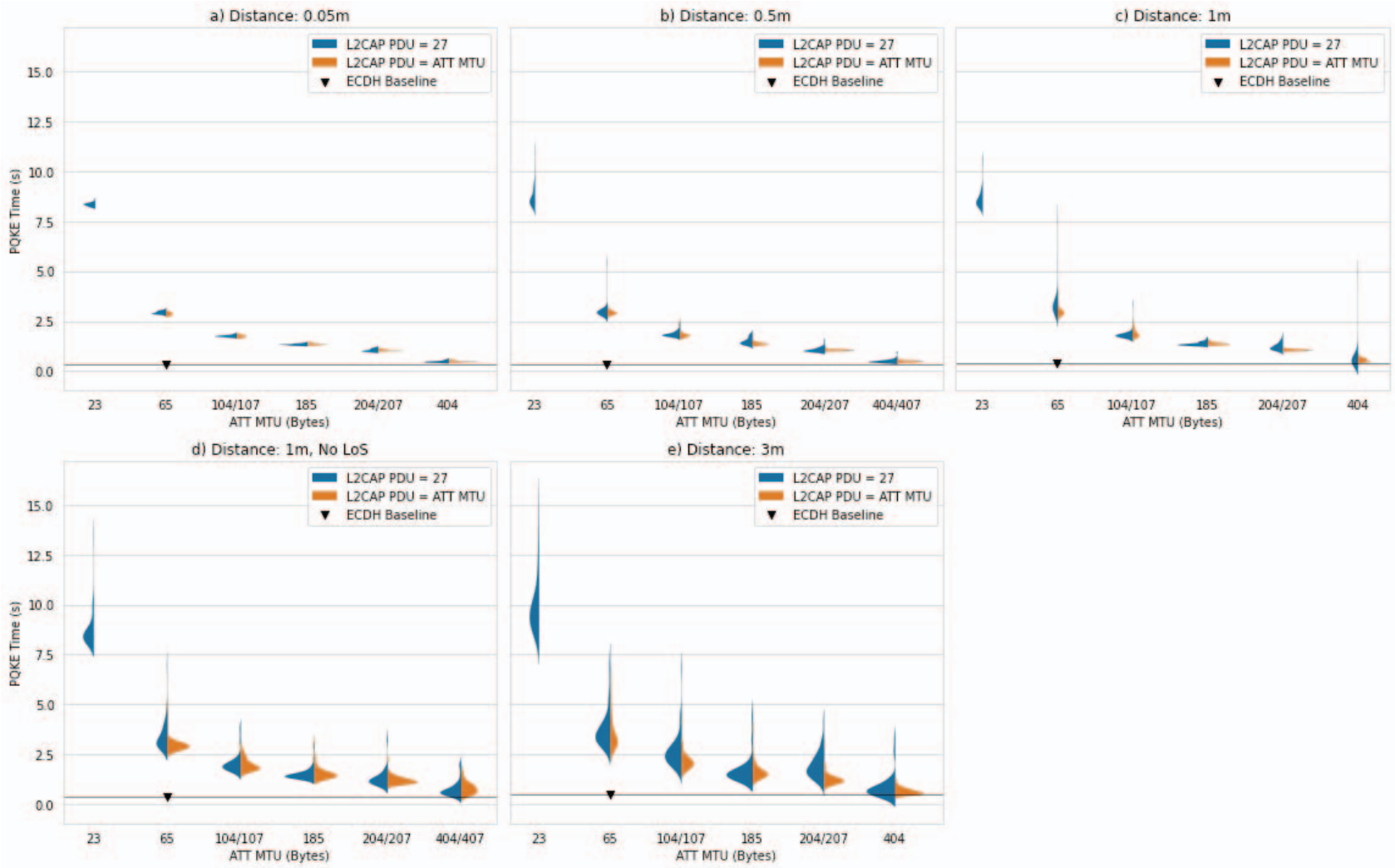

Fig. 4: Violin plots visualizing the distribution of PQKE time with varying ATT MTU, at different Central/Peripheral distances: a) 0.05m; b) 0.50m; c) 1.00m; d) 1.00m without LoS; and e) 3.00m. Horizontal lines indicate average BLE SC pairing time.

Note that figure 4 also compares the performance of PQKE with the standard performance of SC JW pairing, the numerical values of which is presented in Table V.

A. Influence of ATT MTU

Our findings underscore the significant influence of the ATT MTU on the efficiency of the PQKE over BLE. As depicted in Figure 5, there is an inverse correlation between the ATT MTU size and the mean time, $\overline{T}_{pqke}$, required for PQKE. This aligns with our preliminary supposition that unsegmented data transmission is inherently more time-efficient.

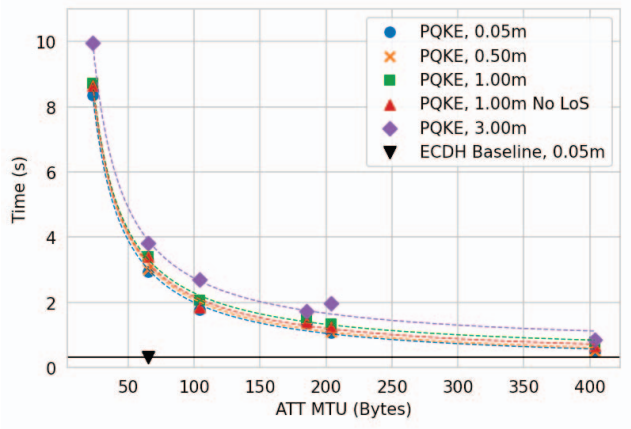

Fig. 5: Plot of average PQKE time against varying ATT MTU, with L2CAP PDU=27, for Central/Peripheral distance = 0.05m, 0.05m, 1.00m, 1.00m without LoS, and 3.00m, respectively. Dashed regression lines show inverse relationship. ECDH pairing time is also shown for comparison.

Upon closer analysis of the data, clear variations in the $\overline{T}_{pqke}$ emerge depending on ATT MTU sizes. Specifically, when considering a short distance=0.05m, $\overline{T}_{pqke}$ measures 8.3673s with an ATT MTU value of 23. This contrasts sharply with a value of 0.4914s when the ATT MTU is 404 bytes. This data suggests that by fine-tuning the ATT MTU, we can reduce the PQKE time by nearly 94%. Similar trends across different configurations can also be observed.

When comparing with SC pairing using JW mode, as illustrated in Figure 4, it's evident that the time cost of PQKE can come close to that of SC pairing with a larger ATT MTU. For instance, at a Central/Peripheral distance of 0.05m, a PQKE using an ATT MTU of 65 bytes averages at 2.9502 seconds, or almost 9 times longer than the traditional SC pairing time (T_{SC}) of 0.3286 seconds. However, with an ATT MTU of 404 bytes, $\overline{T}_{pqke}$ reduces to 0.4914 seconds, marking only a 49.54% increase over T_{SC}. This overhead is calculated for all distances as shown in Table VII. In general, we note that depending on the Central/Peripheral distance, the overhead of optimized $\overline{T}_{pqke}$ in comparison to $\overline{T}_{SC}$ ranges between 48.87% to 103.64%. Although these overheads are still significant when examined on their own, they signal that the latency of future quantum resilient pairing can be decreased to the same order of magnitude as today's SC pairing.

B. Influence of DLE and L2CAP PDU

Further evaluation was undertaken to ascertain if the activation of the DLE feature could enhance PQKE performance by eliminating L2CAP layer segmentation. Contrary to our initial speculation that DLE can substantially boost transmission throughput [16, 28] and by extending PQKE performance, our findings suggest a more intricate relationship.

The data, as shown in Figure 4, reveals that leveraging DLE to bypass L2CAP segmentation isn't a guaranteed recipe for improved PQKE time. While certain configurations, like ATT MTU=65 over a 1m distance, do register a decrease in $\overline{T}_{pqke}$, this isn't a consistent trend. To discern the true impact, we applied T-Tests with the Null Hypothesis H_0 being *eliminating L2CAP segmentation has no effect on* $\overline{T}_{pqke}$, and the Alternative Hypothesis H_A as *eliminating L2CAP segmentation has significant effect on* $\overline{T}_{pqke}$. With a 95% Confidence Interval (CI), it is shown in Table VI that H_A should be accepted in 10 out of 30 configurations, further confirming that there is no evidence that bypassing segmentation on the L2CAP layer can accelerate PQKE performance.

Yet, an intriguing observation is that activating DLE often results in more stable PQKE times. Both Figure 6, which maps the standard deviations of PQKE time, and Figure 4 substantiate this: DLE-enabled configurations typically exhibit tighter distributions and fewer outliers compared to configurations with L2CAP PDU=27.

C. Cost of Computation vs. Cost of Transmission

Examining the overhead of our proposed PQKE reveals a distinction between computation and transmission overhead. This distinction holds significance because, in the context of IoT devices, the computational capacity may improve over time with hardware upgrades (i.e. using a faster microprocessor), whereas the transmission cost is largely dictated by the protocol's specifications.

To illustrate this, we conducted measurements of the average time consumed by the cryptographic functions `Kyber512.KeyGen()`, `Kyber512.Encap()`, and `Kyber512.Decap()`. The cumulative time spent on these operations represents the PQC computation time, denoted as T_c. Remarkably, this computation time remains relatively constant at approximately 0.0345 seconds (as shown in Table VIII), regardless of other parameters. On the other hand, the time dedicated to data transmission, denoted as T_t, can be estimated by subtracting T_c from the average total time $\overline{T}pqke$. Through this analysis, we gain insight into the relative contributions of T_c and $\overline{T}_t$ to the overall $\overline{T}_{pqke}$.

In Table IX, we note that T_c accounts for a maximum of 7.02% of the total time, with ATT MTU=404 bytes and distance is 0.05m. As increasing distance or decreasing ATT MTU would only lead to longer T_t, this ratio can be treated as a conservative estimation, which also underscores the significance of this work: the major source of PQKE performance

		0.05m			0.5m			1m			1m No LoS			3m		
MTU (bytes)	**PDU (bytes)**	$\overline{T}_{pqke}$**(s)**	**SD**	**p**	$\overline{T}_{pqke}$**(s)**	**SD**	**p**	$\overline{T}_{pqke}$**(s)**	**SD**	**p**	$\overline{T}_{pqke}$**(s)**	**SD**	**p**	$\overline{T}_{pqke}$**(s)**	**SD**	**p**
23	27	8.3673	0.0890	-	8.7463	0.5096	-	8.6507	0.4779	-	8.7166	0.8667	-	9.9565	1.4195	-
65	27	2.9502	0.0717	**0.0031**	3.0133	0.3601	0.2159	3.3958	0.7453	**0.0016**	3.4020	0.7030	**0.0015**	3.8196	0.9693	0.1816
65	65	2.8998	0.0744		2.9493	0.0993		3.0170	0.2413		3.0275	0.3964		3.5632	0.7907	
104	27	1.7767	0.0483	0.5002	1.8711	0.1403	0.2285	1.8386	0.2337	0.0913	2.0628	0.4745	0.5585	2.7000	0.8578	**0.0007**
107	107	1.7695	0.0555		1.8360	0.1421		1.9188	0.1967		2.0153	0.3174		2.1858	0.4353	
185	27	1.3536	0.0390	**0.0006**	1.5136	0.1723	**0.0001**	1.3822	0.0882	0.2259	1.5254	0.2825	0.3475	1.7216	0.6845	0.6482
185	185	1.3788	0.0339		1.4057	0.0831		1.3976	0.0505		1.5821	0.3311		1.6677	0.4078	
204	27	1.0629	0.0692	0.5986	1.0904	0.1207	0.1292	1.2428	0.1793	**0.0001**	1.3264	0.4497	0.1569	1.9576	0.6382	**0.0001**
207	207	1.0574	0.0330		1.0626	0.0328		1.0708	0.0344		1.2285	0.2101		1.2995	0.3256	
404	27	0.4914	0.0456	**0.0055**	0.5157	0.0886	0.7423	0.6650	0.6494	0.3913	0.7785	0.3878	0.367	0.8640	0.5756	**0.0116**
404	251	0.5148	0.0303		0.5201	0.0366		0.5831	0.0902		0.8387	0.2974		0.6606	0.1600	

TABLE VI: Average PQKE time, $\overline{T}_{pqke}$, and its standard deviation for each combination of data segmentation and device distance parameters. $p < 0.5$ (in bold) indicates difference in the mean. N = 50 for each configuration.

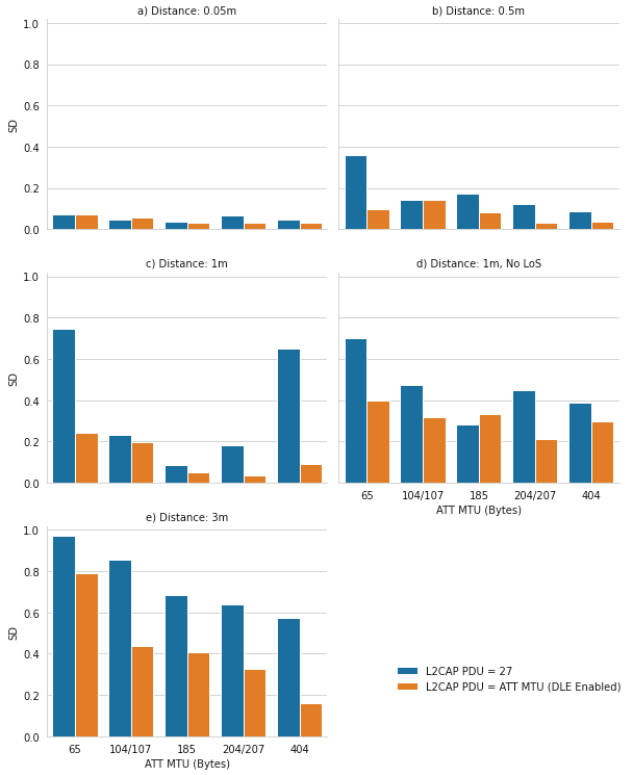

Fig. 6: Plots of Standard Deviation (SD) of PQKE time with varying ATT MTU, at different Central/Peripheral distances: a) 0.05m; b) 0.50m; c) 1.00m; d) 1.00m without LoS; and e) 3.00m.

Distance	**Optmized** $\overline{T}_{pqke}$**(s)**	T_{SC}**(s)**	**Percentage Overhead**
0.05m	0.4914	0.3286	49.54%
0.50m	0.5157	0.3464	48.87%
1.00m	0.665	0.3599	84.77%
1.00m No LoS	0.7785	0.3823	103.64%
3.00m	0.864	0.4702	83.75%

TABLE VII: Percentage overhead of optimized PQKE time (ATT MTU=404 bytes) versus SC pairing time for varying Central/Peripheral distances.

overhead lies in the cost of RF transmission, which merits optimization within the mechanism itself. Conversely, computational costs, largely dependent on microprocessor capabilities, play a comparatively minor role in the overall performance, and should be treated as a less important optimization target.

Algorithm	**Computed By**	**Time (s)**
`Kyber512.KeyGen()`	Peripheral	0.0144
`Kyber512.Encap()`	Central	0.0004
`Kyber512.Decap()`	Peripheral	0.0197
T_c		**0.0345**

TABLE VIII: Estimated time spent in computing cryptographic functions of Kyber-512.

MTU	**PDU**	$\overline{T}_{pqke}$ (s)	T_c (s)	$T_c/\overline{T}_{pqke}$
23	27	8.3673	0.0345	0.41%
65	27	2.9502		1.17%
65	65	2.8998		1.19%
104	27	1.7767		1.94%
107	107	1.7695		1.95%
185	27	1.3536		2.55%
185	185	1.3788		2.50%
204	27	1.0629		3.25%
207	207	1.0574		3.26%
404	27	0.4914		7.02%
404	251	0.5148		6.70%

TABLE IX: Ratio of estimated T_c against $\overline{T}_{pqke}$ at close proximity (distance=0.05m). The computation cost of Kyber-512 is not a major source of PQKE latency.

VI. Conclusion and Future Work

In the rapidly evolving landscape of cybersecurity, the advent of quantum computers presents novel challenges for RF-based technology and IoT security in general. Our exploration into the nuances of PQKE for BLE unveils insights that pave the way for future advancements.

Our study underscores the pivotal role of ATT MTU optimization, which can significantly improve PQKE time, with observed accelerations of up to 94%. When ATT MTU is maximized, the temporal cost of PQKE approaches that of BLE SC pairing with a variable overhead between 48.87% and 103.64%, depending on the Central/Peripheral distance. These findings suggest that quantum resilient pairing for BLE

is a feasible possibility using existing technology. Furthermore, we note that eliminating L2CAP segmentation isn't a universal solution to expedite PQKE. While it may not always bolster speed, it imparts a degree of stability to the time cost of PQKE, which may be useful for performance-sensitive applications. Finally, we found that a large part of the performance overhead in PQKE is attributed to the transmission cost, rather than the cryptographic computations themselves. This nuance redirects our attention to transmission mechanisms as key areas for enhancement.

In light of these findings and the looming quantum threats, we outline several recommendations for BLE's Post-Quantum security:

- **PQC Migration.** The trajectory towards future-proof cybersecurity mandates a shift to PQC. It's important to revisit BLE's pairing mechanisms and integrate PQC for key establishment.
- **Kyber KEM.** The Kyber KEM emerges as a promising tool for preparing BLE against the quantum threats. We advocate its potential integration into subsequent versions of the Bluetooth Core Specification.
- **ATT MTU Considerations.** Designs for quantum resilient BLE pairing should consider maximizing ATT MTU when possible, due to extended data exchange. This is especially important for applications where latency is at a premium.
- **DLE Feature Utilization.**: For devices equipped with the DLE feature, its deployment can mitigate PQKE latency variations, ensuring a consistent and reliable performance.

In this study, we concentrated on analyzing the pairing time/latency as a key performance metric and provided recommendations based on these findings. Another critical consideration for IoT applications is energy efficiency, particularly for lightweight IoT nodes operating on limited energy sources, such as batteries and energy harvesting. Although operation time is linked to overall energy consumption, it is important to note that transmitting data more rapidly can increase power usage. Consequently, it becomes essential for future investigations to explore the energy efficiency of utilizing larger ATT MTU sizes. Furthermore, our research observed that activating DLE may not result in reduced pairing time, contrary to expectations of its beneficial effect on data transmission rates. The underlying reasons for this observation merit further exploration to understand the dynamics fully.

In conclusion, with the advancements in quantum computing, it's important to proactively update network protocols and infuse them with quantum resilience. The results and suggestions from this study provide a practical roadmap for enhancing the BLE framework to ensure its security, compatibility and efficiency in a Post-Quantum era.

REFERENCES

[1] Gorjan Alagic et al. *Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process*. https://doi.org/10.6028/NIST.IR.8240. 2019.

[2] Gorjan Alagic et al. *Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process*. https://doi.org/10.6028/NIST.IR.8413-upd1. 2022.

[3] Armands Ancans et al. "Bluetooth low energy throughput in densely deployed radio environment". In: *2019 23rd International Conference Electronics*. IEEE. 2019, pp. 1–5.

[4] Daniel J Bernstein and Tanja Lange. "Post-quantum cryptography". In: *Nature* 549.7671 (2017), pp. 188–194.

[5] Joppe Bos et al. "CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM". In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2018, pp. 353–367.

[6] Jessica Bozhko et al. "Performance Evaluation of Quantum-Resistant TLS for Consumer IoT Devices". In: *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*. IEEE. 2023, pp. 230–235.

[7] Kevin Bürstinghaus-Steinbach et al. "Post-quantum tls on embedded systems: Integrating and evaluating kyber and sphincs+ with mbed tls". In: *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*. 2020, pp. 841–852.

[8] Kavita Char. *Internet of things system design with integrated wireless MCUs*. Tech. rep. Silicon Labs, ARM, Tech. Rep, 2015.

[9] Alexandre Augusto Giron. *Migrating Applications to Post-Quantum Cryptography: Beyond Algorithm Replacement*. Cryptology ePrint Archive, Paper 2023/709. https://eprint.iacr.org/2023/709. 2023. URL: https://eprint.iacr.org/2023/709.

[10] Ruben Gonzalez and Thom Wiggers. "KEMTLS vs. Post-quantum TLS: Performance on Embedded Systems". In: *International Conference on Security, Privacy, and Applied Cryptography Engineering*. Springer. 2022, pp. 99–117.

[11] Lov K Grover. "Quantum mechanics helps in searching for a needle in a haystack". In: *Physical review letters* 79.2 (1997), p. 325.

[12] David Joseph et al. "Transitioning organizations to post-quantum cryptography". In: *Nature* 605.7909 (2022), pp. 237–243.

[13] Matthias J. Kannwischer et al. *PQM4: Post-quantum crypto library for the ARM Cortex-M4*. https://github.com/mupq/pqm4.

[14] Adarsh Kumar et al. "Securing the future internet of things with post-quantum cryptography". In: *Security and Privacy* 5.2 (2022), e200.

[15] Somayya Madakam et al. "Internet of Things (IoT): A literature review". In: *Journal of Computer and Communications* 3.05 (2015), p. 164.

[16] Zach Michel. *Maximizing BLE Throughput Part 3: Data Length Extension (DLE)*. https://punchthrough.com/maximizing-ble-throughput-part-3-data-length-extension-dle-2/. Accessed: 2024-2-18. 2019.

[17] Michele Mosca and Marco Piani. *2022 quantum threat timeline report*. Tech. rep. Global Risk Institute, 2022.

[18] William Newhouse et al. *Migration to Post-Quantum Cryptography: Preparation for Considering the Implementation and Adoption of Quantum Safe Cryptography (Initial Preliminary Draft)*. Tech. rep. National Institute of Standards and Technology, 2023.

[19] NIST. *Comments Requested on Three Draft FIPS for Post-Quantum Cryptography*. Accessed 2023-9-22. Aug. 2023. URL: https://www.nccoe.nist.gov/crypto-agility-considerations-migrating-post-quantum-cryptographic-algorithms.

[20] NIST. *Post-Quantum Cryptography*. Accessed 2024-2-18. Jan. 2017. URL: https://csrc.nist.gov/projects/post-quantum-cryptography.

[21] David Ott, Christopher Peikert, and other workshop participants. *Identifying Research Challenges in Post Quantum Cryptography Migration and Cryptographic Agility*. 2019. arXiv: 1909.07353.

[22] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (Aug. 2018), p. 79. ISSN: 2521-327X. DOI: 10.22331/q-2018-08-06-79. URL: https://doi.org/10.22331/q-2018-08-06-79.

[23] PC Sajimon, Kurunandan Jain, and Prabhakar Krishnan. "Analysis of post-quantum cryptography for internet of things". In: *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE. 2022, pp. 387–394.

[24] Maximilian Schöffel et al. "On the Energy Costs of Post-Quantum KEMs in TLS-based Low-Power Secure IoT". In: Association for Computing Machinery, Inc, Oct. 2021, pp. 158–168. ISBN: 9781450383547. DOI: 10.1145/3450268.3453528.

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

[26] Bluetooth SIG. *Bluetooth Core Specification*. Jan. 2023. URL: https://www.bluetooth.com/specifications/specs/core-specification-5-4/.

[27] C. C. Sobin. "A Survey on Architecture, Protocols and Challenges in IoT". In: *Wireless Personal Communications* 112.3 (2020), pp. 1383–1429. ISSN: 1572-834X. DOI: 10.1007/s11277-020-07108-5. URL: https://doi.org/10.1007/s11277-020-07108-5.

[28] Texas Instruments. *LE Data Length Extension (DLE)*. https://software-dl.ti.com/lprf/simplelink_cc2640r2_latest/docs/blestack/ble_user_guide/html/ble-stack-3.x/data-length-extensions.html. Accessed: 2024-02-18. 2016.

[29] Jacopo Tosi et al. "Performance Evaluation of Bluetooth Low Energy: A Systematic Review". In: *Sensors* 17.12 (2017). ISSN: 1424-8220. DOI: 10.3390/s17122898. URL: https://www.mdpi.com/1424-8220/17/12/2898.

[30] Sean Turner. "Transport Layer Security". In: *IEEE Internet Computing* 18.6 (2014), pp. 60–63.

[31] Yousaf Bin Zikria et al. "Internet of Things (IoT) operating systems management: Opportunities, challenges, and solution". In: *Sensors* 19.8 (2019), p. 1793.