PQ-OTSec: A Post-Quantum Secure Key Exchange Framework for Operational Technology Protocols

Nasir Bala Muhammad*, Abubakar Sadiq Sani*, Don Yuan!, Georgia Sakellari*, and George Loukas*
* School of Computing and Mathematical Sciences, University of Greenwich, London, United Kingdom
Email: {N.B.Muhammad, S.Sani, G.Sakellari, and G.Loukas}@greenwich.ac.uk
! School of Electrical and Information Engineering, The University of Sydney, Sydney, Australia
Email: Dong.Yuan@sydney.edu.au

***Abstract*—Security mechanisms for Operational Technology (OT) protocols like Modbus/TCP Security, DNP3-SAv6 and Profinet Security protect the underlying protocols that govern how industrial automation systems communicate. However, these solutions are dependent on classical cryptography and lack the agility to withstand the emerging threat posed by quantum computing, which breaks or weakens classical cryptography. Furthermore, the increasing sophistication of classical cyber-attacks on industrial control systems, as evidenced by the recent Polish power grid incident of December 2025, shows the growing capabilities of adversaries. When considered alongside the anticipated capabilities of quantum computing, this necessitates the development of robust and future-resilient security mechanisms for OT environments. In this paper, we present PQ-OTSec, a protocol-agnostic, post-quantum (PQ) secure overlay that enables mutual authentication and secure key exchange without modifying the existing protocol structure. Our framework employs NIST-standardised ML-DSA for authentication and ML-KEM for key establishment and achieves confidentiality, integrity, availability, and authenticity in PQ settings. We assess the security of our framework using the AVISPA tool and demonstrate its applicability by overlaying it onto Modbus/TCP. In our experiment, the framework shows better performance than Modbus/TCP Security, making it suitable for deployment in OT environments. We finally use PQ-OTSec to mitigate the vulnerabilities exploited in the Polish power grid attack, as well as protect against future quantum threats.**

***Index Terms*—operational technology protocols, post-quantum cryptography, communication security, quantum computing**

I. Introduction

Operational Technology (OT) refers to devices and applications that enable real-time control, management, and monitoring of physical processes in power grids, industrial automation systems, and energy distribution networks [1]. The reliability and efficiency of these systems are highly dependent on communication protocols such as Modbus, Distributed Network Protocol 3 (DNP3), and Profinet, which facilitate the exchange of critical process control data. Historically, these protocols were designed with minimal cybersecurity considerations and prioritised interoperability and reliability over security. As a result, most OT protocols lack native authentication and encryption capabilities. With the increasing convergence of OT and Information Technology (IT) environments, these protocols are now exposed to a wide range of cybersecurity threats affecting confidentiality, integrity, authenticity and availability [2].

The Polish power grid attack [3] discovered in December 2025 is an example of such cyber threats resulting from IT/OT convergence. In this incident, attackers gained access to the OT network through an unsecured VPN. Devices within the OT network communicated via DNP3 and IEC 101 protocols without additional security protections. As a result, the attackers were able to move laterally across the network without restriction to initiate destructive activities. They deployed modified device firmware, injected malicious commands, and wiped device files, which ultimately caused loss of communication on parts of the grid. Although this attack was detected early and did not compromise the overall stability of the Polish power grid, the lack of secure protocol configurations and strong authentication gave the attackers unrestricted access to carry it out.

The exposure of OT protocols to cyber threats has driven the development of security mechanisms such as Modbus/TCP Security [4], DNP3 Secure Authentication (DNP3-SAv6) [5], and Profinet Security [6]. However, these solutions rely heavily on classical cryptographic primitives like RSA, Diffie-Hellman (DH), elliptic curve cryptography (ECC) and AES, which are vulnerable to the emerging threats posed by quantum computing [2]. In particular, Shor's algorithm [7] can efficiently solve the integer factorisation and discrete logarithm problems, thereby breaking asymmetric schemes such as RSA, DH, and elliptic curve Diffie-Hellman (ECDH). Additionally, Grover's algorithm [8] provides a quadratic speedup in brute-force attacks on symmetric ciphers such as AES and SHA. Although operations-grade quantum computers are not yet practical, and OT devices typically have operational lifespans of 15-20 years [1], many devices deployed today are likely to remain in use into the era of practical quantum technologies. Consequently, the absence of post-quantum cryptographic protections in OT protocols poses a significant risk to the safety, availability, and operational integrity of these systems.

Existing standards have proposed procedures for securing OT environments through high-level frameworks like IEC 62443-3-3 [9], IEC 62351-5 [10], and NIST SP 800-82r3 [1]. However, these approaches generally focus on system-level policies and post-implementation controls, with limited attention to protocol-layer deficiencies and their adaptation to post-quantum requirements. Furthermore, recent research on PQC implementations [11]-[14] has largely focused on IT

TABLE I
CLASSICAL AND POST-QUANTUM CRYPTOGRAPHIC CAPABILITIES AND LIMITATIONS OF RELATED SOLUTIONS

Solution	Classical Cryptographic Capabilities				Post-quantum Cryptographic Capabilities				OT Applicability
	Confidentiality	Integrity	Availability	Authentication	Confidentiality	Integrity	Availability	Authentication	
[4]	✓	✓	✓	✓	✗	✗	✗	✗	✓
[5]	✓	✓	✓	✓	✗	✗	✗	✗	✓
[6]	✓	✓	✓	✓	✗	✗	✗	✗	✓
[11]	✓	✓	✓	✓	✓	✓	✓	✓	✗
[12]	✗	✗	✗	✗	✗	✗	✗	✗	✓
[14]	✓	✓	✓	✓	✓	✓	✗	✓	✓

systems, IoT environments, or specific protocols and relies on non-standardised or hybrid approaches that do not align with emerging NIST PQC standards, which include the Module Lattice-based Digital Signature Algorithm (ML-DSA) [15], and the Module Lattice-based Key Encapsulation Mechanism (ML-KEM) [16]. As a result, there remains a lack of protocol-agnostic frameworks that integrate standardised PQC primitives into OT protocols while preserving compatibility with legacy systems.

In this paper, we propose PQ-OTSec, a protocol agnostic post-quantum secure communication framework for OT protocols that operates without modifying the underlying protocol structure. PQ-OTSec introduces a lightweight security overlay that satisfies the post-quantum security goals of confidentiality, integrity, availability, and authenticity using NIST-standardised ML-DSA (FIPS 204) for authentication and ML-KEM (FIPS 203) for secure key establishment. The main contributions of this work are as follows: i) We design PQ-OTSec, a protocol-agnostic post-quantum secure overlay for OT protocols; ii) We integrate ML-DSA and ML-KEM to achieve quantum-resistant mutual authentication and key exchange; iii) We formally verify the security of the proposed framework using the Automated Validation of Internet Security Protocols and Applications (AVISPA) [17] under the Dolev-Yao threat model; iv) We demonstrate the applicability of PQ-OTSec through integration with Modbus/TCP using *liboqs* [18] and *libmodbus* [19] to produce a dedicated secure communication protocol, PQ-ModSec; and v) We use our framework to mitigate vulnerabilities exploited in the Polish power grid attack, as well as protect against future quantum threats.

II. RELATED WORK

Classical security solutions addressing legacy security weaknesses for OT protocols follow established standards, each with distinct advantages and trade-offs. Modbus/TCP Security [3] incorporates a TLS 1.2 [21] wrapper to protect communications between devices. TLS 1.2 relies on X.509 certificates and classical primitives such as RSA, ECDSA, and ECDH for authentication and key exchange [16]. While a standardised and widely adopted security framework, TLS introduces operational complexity due to its dependence on certificate authorities (CA) and associated revocation mechanisms, which can affect availability in constrained or isolated OT networks. Moreover, TLS 1.2 does not provide post-quantum security guarantees. All asymmetric ciphers used in TLS (RSA, DH, ECDH and ECDSA) are rendered insecure by Shor's algorithm, while symmetric primitives like (AES and SHA) are weakened by Grover's algorithm [20].

Similarly, DNP3-SAv6 [4], as aligned to IEC 62351-5 standard [10], enhances protocol security through challenge-response mechanisms and supports the use of symmetric keys and ECC for authentication, with optional encryption. While it improves resilience for the DNP3 protocol against unauthorised access and command injection, its reliance on classical cryptographic primitives renders it vulnerable to quantum threats, particularly those exploiting Shor's algorithm. In addition, although DNP3-SAv6 can leverage certificate-based infrastructure, such as X.509, in practice it often depends on pre-shared keys by a centralised key distribution authority at device enrolment. This introduces complexities and potential risks during key provisioning if provisioning procedures are not adequately protected.

Profinet Security [6] references IEC 62443-3-3 [9] and extends the Profinet protocol by incorporating security classes 1-3 that provide integrity protection, authentication and optional encryption. These mechanisms are implemented using established classical cryptographic techniques under the TLS framework, rather than defining a dedicated authentication and key exchange mechanism. While this approach enhances the security of Profinet, it introduces dependencies on certificate management infrastructures and classical cryptographic primitives, which do not provide post-quantum security protection.

Beyond protocol-specific enhancements, several studies have explored the implementation of PQC in IoT and other resource-constrained environments. The research in [11] implemented the post-quantum digital signature algorithm, Dilithium (a precursor to ML-DSA) on ESP32 microcontrollers for secure IoT applications in heart rate and blood monitoring. Despite this success, OT environments impose stricter latency requirements and deterministic communication cycles, which are not typically encountered on general embedded systems. [12] proposed a multiphase quantum-resistant framework for securing SCADA communications by combining Supersingular Isogeny-based Key Encapsulation (SIKE), quantum key distribution, and ASCON lightweight cryptographic primitive. However, NIST's official withdrawal of the SIKE primitive due to discovered vulnerabilities [13] limits the framework's practical applicability. [14] integrated selected post-quantum cryptographic primitives (Dilithium, FALCON, SPHINCS+) into OPC UA industrial protocol using

both hybrid (PQC + classical cryptography) and non-hybrid approaches. While effective for OPC UA, the solution does not generalise to other protocols and is implemented using non-standardised versions of the algorithms. Table I summarises the reviewed solutions, their classical and PQ cryptographic capabilities, and their application to OT environment.

Considering the above discussion, three consistent research gaps emerge. First, existing OT protocol security specifications and standards rely heavily on classical cryptography, which is vulnerable to quantum attacks. Second, although prior studies have explored PQC IoT and industrial contexts, they have not been specifically tailored to OT environments in critical infrastructure, and no reference implementation currently integrates standardised ML-DSA or ML-KEM into OT protocols. Third, most deployed classical security frameworks depend on external trust infrastructures, such as CAs or key servers, which are often impractical for the operational realities of connected OT devices. These gaps motivate the design of our PQ-OTSec framework, which we present in the following section.

III. The PQ-OTSec Framework

The PQ-OTSec framework defines a set of cryptographic operations and key encapsulation steps to satisfy mutual authentication and key secrecy. It utilises dual cryptographic schemes: ML-DSA [15] for mutual authentication and ML-KEM [16] for key exchange. Both schemes are based on lattice cryptography, where the security relies on the hardness of structured problems, Module-Learning-With Errors (M-LWE) and Module-Short-Integer-Solution (M-SIS), as opposed to classical cryptography, where the hardness is in the integer factorisation or the discrete logarithm problems. ML-KEM domain parameters is instantiated over the M-LWE problem, while ML-DSA relies on both M-SIS and M-LWE, defined over a polynomial ring $R_q = \mathbb{Z}_q[x]/(x^{256}+1)$. The prime modulus q is globally set to 8380417 for ML-DSA and 3329 for ML-KEM, in accordance with standardised parameter sets aligned with NIST security levels, 1, 3, and 5. Parameter selection is performed at runtime. For the rest of the document, we denote ML-DSA public and private key pairs as pk, sk, and ML-KEM encapsulation and decapsulation keys as ek, dk respectively.

A. Device Enrolment and Verification

An OT device in PQ-OTSec is identified by an ID_i within a set of identities. The framework also maintains a trusted registry TR, which is a subset of identities mapped to generated ML-DSA public keys (i.e. $(ID, pk) \in TR$). All devices operating within the framework have synchronised, real-time access to the registry. This approach eliminates the need for CAs and ensures key trust across the network. ML-DSA keys are long-term. For simplicity, we assume TR is secure. Device enrolment into the PQ-OTSec framework initiates the following operations:

1) **Parameter selection.** Select ML-DSA-(kl) parameter set, where (k, ℓ) are the dimensions of matrix A that defines the lattice. The matrix dimensions in ML-DSA are either $(4, 4)$, $(6, 5)$, or $(8, 7)$, each corresponding to the NIST security levels respectively.
2) **Select random seed value.** Select a 32-byte random seed value $\xi \in \{0, 1, \ldots, 255\}$.
3) **Seed expansion.** Convert k and ℓ from integers to base-256 and compute the function $H(\xi \,\|\, k \,\|\, \ell, 128)$ to extract 128 bytes, where H is the SHAKE256 hash function in byte-output mode. The 128-byte output is divided into three seeds: a 32-byte public seed ρ, a 64-byte private seed ρ', and another 32-byte private seed K used during signing.
4) **Generate matrix A and vectors s_1 and s_2.** Using ρ, generate matrix $\mathbf{A} \in R_q^{k \times \ell}$. Using ρ', generate two small polynomial vectors $s_1 \in R^{\ell}$ and $s_2 \in R^k$ whose coefficients are in the range $[-\eta, \eta]$, where the private key is sampled.
5) **Compute the public vector** t. Compute $t = A \circ s_1 + s_2$. Then compress t by dropping the least-significant bit from each coefficient to obtain t_1 and encode the least-significant bit of each coefficient to obtain t_0.
6) **Construct public and private key pair** (pk, sk). Encode (ρ, t_1) as public key pk, and store (ID, pk) to the set TR. Compute $H(pk, 64)$ to obtain a 64-byte hash value tr of the public key. Encode $(\rho, K, tr, s_1, s_2, t_0)$ and store it as private key sk.

After devices are enrolled on the network and all identities are registered in TR along with their ML-DSA pk, an Initiator of a session (ID_A) can verify the public key of a Responder (ID_B) by checking that $(ID, pk) \in TR$. If the check is successful, ID_A proceeds to establish a shared key with ID_B. Otherwise, the session terminates. Both ID_A and ID_B execute this verification step, and it must be successful for the communication to continue.

B. Key Exchange

After verifying both devices and their corresponding pk's exist in TR, ID_A proceeds to generate encapsulation and decapsulation key pairs (ek, dk). To ensure perfect forward secrecy, ek and dk are ephemeral and computed by the initiator at the beginning of each session and subsequently discarded at the end. The key exchange in PQ-OTSec initiates the following operations:

1) **Parameter selection.** Select ML-KEM-x parameter set, where x corresponds to target security strength. ML-KEM-512 gives approximated equivalence to 128-bit classical cryptographic security, -768 gives 192-bit security, and -1024 gives 256-bit security. It also determines the dimensions of the matrix A that defines the lattice.
2) **Select random seed values.** Select two random 32-byte seed values $d, z \in \{0, 1, \ldots, 255\}$. z is used in computing the implicit reject value.
3) **Seed expansion**. Compute the function $G(d \,\|\, k)$ to expand the $32+1$ bytes into two 32-byte pseudorandom seeds ρ and σ, where G is a SHA3-512 hash function.

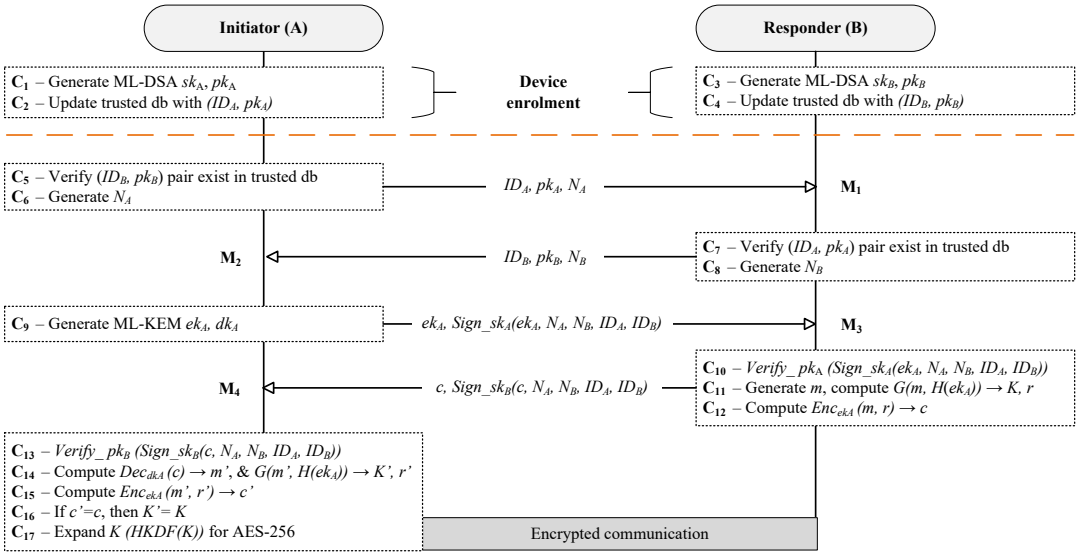

Fig. 1. PQ-OTSec framework high-level protocol flow.

ρ is a public seed used to derive the matrix, and σ is used to sample small pseudorandom vectors.

4) **Generate matrix** A **and vectors** s and e. Using ρ, generate matrix $\mathbf{A} \in (\mathbb{Z}_q^{256})^{k \times k}$. Using σ, generate secret vector $\mathbf{s} \in (\mathbb{Z}_q^{256})^k$ and noise vector $\mathbf{e} \in (\mathbb{Z}_q^{256})^k$.
5) **Compute noisy linear system** t. Compute $t = A \circ s + e$.
6) **Generate encapsulation and decapsulation key pair** (ek, dk). Compute a byte-encoding of t and s to obtain t' and s', then compute $(t' \,\|\, \rho)$ as encapsulation key ek. Finally, compute $(s' \,\|\, ek \,\|\, H(ek) \,\|\, z)$, where H is a SHA3-256 function, and store it as decapsulation key dk.
7) **Key encapsulation.** ID_A sends ek to ID_B, and upon receipt, ID_B generates a 32-byte random message $m \in \{0, 1, \ldots, 255\}$, and then computes $G(m \,\|\, H(ek))$ to obtain a 64-byte output. The first 32 bytes are retained as the shared secret key K, while the remaining 32 bytes are used as randomness r to create a ciphertext c. ID_B computes $\mathrm{Enc}_{ek}(m, r)$ to obtain c, where $\mathrm{Enc}(\cdot)$ is an IND-CPA secure encryption algorithm inherent in ML-KEM. c is then sent back to ID_A.
8) **Key decapsulation.** At receipt of c, ID_A runs $\mathrm{Dec}_{dk}(c)$ to obtain m', where $\mathrm{Dec}(\cdot)$ is an internal decryption algorithm inherent in ML-KEM. Afterwards, it computes $G(m' \,\|\, H(ek))$ to obtain a 64-byte output. The first 32 bytes are retained as the candidate shared secret key K', while the remaining 32 bytes are used as randomness r'. ID_A then computes $\mathrm{Enc}_{ek}(m', r')$ in the same manner as done in encapsulation step to obtain c'. If $c' = c$, then $K' = K$. This confirms that the responder has derived the same shared secret key. Otherwise, compute and return the value $\bar{K} = J(z \,\|\, c)$, where J is a SHAKE256 hash function and z is the implicit rejection value derived above. The return of $\bar{K}$ ensures that rejection is not revealed to an observer.

The steps above showcase the key exchange process of the PQ-OTSec framework, which ensures mutual authentication, key secrecy, and perfect forward secrecy. The shared secret obtained can then be used as an input to a HKDF to produce keys for use with quantum-resistant symmetric ciphers such as AES-256, while considering the low resource and power constraints of OT devices. A high-level protocol flow of PQ-OTSec framework is presented in Fig. 1. C_i indicates device operations while M_i indicates messages sent between devices to establish the key.

IV. Formal Security Verification

In this section, we specify and verify the security of our proposed PQ-OTSec framework using AVISPA [17], which is a tool for automated cryptographic security analysis and validation. We implement PQ-OTSec using the High-Level Protocol Specification Language (HLPSL) to model the roles of the communicating devices for mutual authentication, secure key exchange, and attacker knowledge. For simplicity, we assume device enrolment has already occurred securely.

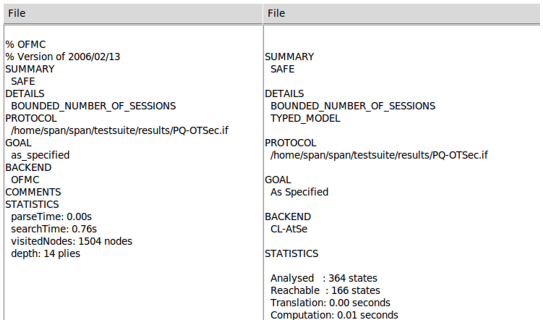

Fig. 2. AVISPA simulation results using OFMC and CL-AtSe backends.

We only model the mutual authentication and key exchange between two devices ID_A and ID_B. We follow the Dolev-Yao threat model to describe the capabilities of an adversary who can eavesdrop, intercept, and manipulate data transmitted between devices. In our implementation, we used the On-the-Fly Model Checker (OFMC) and Constraint Logic-based Attack Searcher (CL-AtSe) modules available in AVISPA to check for security attacks. Details of the protocol are as follows: Step 1: $ID_A \rightarrow ID_B = (ID_A, pk_A, N_A)$; Step 2: $ID_B \rightarrow ID_A = (ID_B, pk_B, N_B)$; Step 3: $ID_A \rightarrow ID_B = (ek_A, \text{Sign}_{sk_A}(N_A, N_B, ID_A, ID_B))$; Step 4: $ID_B \rightarrow ID_A = (c, \text{Sign}_{sk_B}(c, N_A, N_B, ID_A, ID_B))$.As shown in Fig. 2, the simulation results from the OFMC and CL-AtSe modules confirm that PQ-OTSec is resilient to replay and man-in-the-middle attacks under the Dolev-Yao model, and is therefore secure within this threat model.

V. Case Study

A. Architecture and Setup

To demonstrate the applicability of the proposed PQ-OTSec framework, we integrate PQ-OTSec with the Modbus/TCP protocol to obtain PQ-ModSec. The integration of PQ-OTSec follows a non-intrusive overlay approach that preserves the original Modbus protocol structure and establishes a secure communication channel beneath the application layer. PQ-OTSec acts as a secure socket abstraction which transparently encrypts and authenticates all Modbus messages exchanged between client and server.

Our proof-of-concept integration is based on the open-source C libraries of *liboqs* [18] for implementing ML-DSA and ML-KEM primitives, and *libmodbus* [19] for Modbus/TCP communication. We establish a custom wrapper function that i) establishes a PQ-OTSec handshake between client and server; ii) authenticates devices using ML-DSA signatures; iii) derives a shared key using ML-KEM; and iv) encrypt and decrypt Modbus payloads using AES-256. The secure channel is initialised before any Modbus request is transmitted. PQ-ModSec architecture is illustrated in Fig. 3.

B. PQ-ModSec Protocol

The PQ-ModSec overlay protocol occurs between a Modbus client (Initiator) and server (responder) and proceeds as follows: i) participants exchange identities and verify each other's public keys against the trusted registry; ii) initiator generates an ephemeral ML-KEM key pair and transmits the encapsulation key to the responder; iii) the responder encapsulates a randomly generated secret using the received key and returns the resulting ciphertext to the initiator; and iv) the initiator decapsulates the ciphertext to derive the shared session key and verify correctness by recomputing the ciphertext. Mutual authentication occurs when both parties sign nonces, identities and exchanged values using ML-DSA, which prevents man-in-the-middle attacks. Upon successful verification, both parties derive the shared symmetric key, which is used to encrypt subsequent Modbus communication.

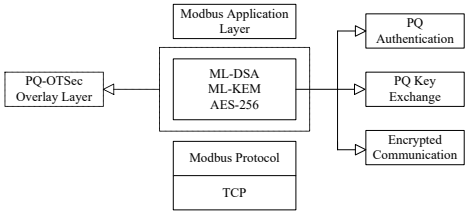

Fig. 3. High-level architecture of PQ-ModSec.

C. Performance Considerations

PQ-ModSec introduces computational and communication overhead compared to classical Modbus/TCP due to the larger signature and key sizes for ML-DSA and ML-KEM, as well as the additional latency of the post-quantum handshake. However, these overheads are incurred only during session establishment. Once a secure channel is established, communication proceeds efficiently using AES, as is the case for Modbus/TCP Security. Signature and key sizes are outlined in [15] and [16] respectively.

Our experimental setup for performance measurement was conducted on an Ubuntu 25.04 VM with Intel i7 CPU @ 1.70 GHz and 4GB RAM. It is important to note that this a non-isolated environment and measurement noise is expected. All reported results represent the average execution time over multiple iterations of each operation within a 3-second period. We measured the execution time of the core operations for all parameter sets of ML-DSA and ML-KEM. The output of our performance test for each case is summarised in Fig. 4. For this experiment, to manage paper-length constraint, we focus on the lowest parameter sets (ML-DSA-44 and ML-KEM-512).

In PQ-ModSec, a complete handshake requires i) two ML-DSA signature operations ($379.554\,\mu s \times 2$), ii) two ML-DSA

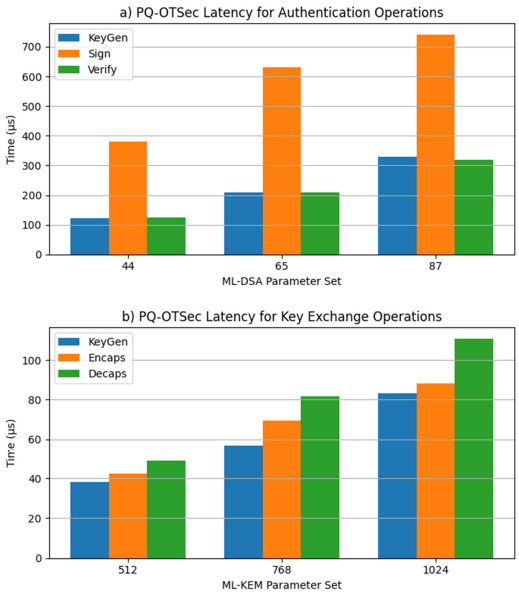

Fig. 4. Benchmark Operations Latency for a) Authentication and b) Key Exchange

verification operations ($125.108\,\mu s \times 2$), iii) one ML-KEM key generation ($37.947\,\mu s$), iv) one encapsulation ($42.468\,\mu s$) and v) one decapsulation ($48.676\,\mu s$). The average latency sums up to approximately $1138.4\,\mu s$. To evaluate relative performance, we implemented Modbus/TCP Security handshake using OpenSSL [22] under the same experimental conditions as PQ-ModSec. Modbus/TCP Security uses `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256` as the minimum cipher suite. Using a self-signed certificate, we benchmarked the average latency over a 3-second period, and was measured at approximately $1380\,\mu s$. It should be noted that the use of a self-signed certificate excludes CA validation overhead, which will increase Modbus/TCP Security latency in production deployment. These results show that PQ-ModSec offers better performance compared to Modbus/TCP Security, with approximately 17.5% lower handshake latency.

D. Mitigating the Polish Power Grid Attack

The Polish power grid attack exploited the absence of access control, authentication, protocol-level security mechanisms, as well as implicit trust within the OT network. The existing protocols are unable to withstand the attack. By enforcing mutual authentication, message integrity, and secure session establishment for all device interactions, PQ-OTSec addresses these vulnerabilities. Even if the attacker gains access to the network through an unsecured VPN, they cannot establish valid communication sessions or inject malicious commands without valid cryptographic credentials. Furthermore, PQ-OTSec eliminates implicit trust between devices by ensuring each session is independently authenticated and secured. This prevents unrestricted lateral movement within the network. Additionally, the use of ML-DSA and ML-KEM in PQ-OTSec ensures that the grid remains secure against future adversaries with quantum capabilities, thereby providing long-term resilience for OT systems.

VI. Conclusion and Future Work

In this paper, we introduced PQ-OTSec, a protocol-agnostic, post-quantum secure communication framework for OT protocols to protect OT environments in critical infrastructure against the emerging threat of quantum computing. We combined ML-DSA and ML-KEM to provide quantum-resistant mutual authentication and key exchange within an overlay architecture that does not modify underlying OT protocol structure. Through formal verification with AVISPA and practical integration to Modbus/TCP, we showed that PQ-OTSec achieves confidentiality, integrity, availability and authenticity in post-quantum threat environment while remaining deployable in legacy systems. We compared the performance of our protocol against Modbus/TCP Security and the results shows comparable performance outputs, with our protocol having slightly lower handshake latency. We utilise this solution to mitigate the Polish power grid attack. In future work, we will evaluate PQ-OTSec on OT hardware by conducting this experiment under real conditions to provide and an optimal balance between security and usability. Additionally, we will extend PQ-OTSec to other OT protocols like DNP3 and Profinet to strengthen its protocol-agnostic applicability.

References

[1] K. Stouffer, *et al.,* "Guide to operational technology (OT) security," *NIST Special Publication 800-82r3,* National Institute of Standards and Technology, Gaithersburg, MD, USA, 2023, doi: 10.6028/NIST.SP.800-82r3.

[2] M. J. D. Vermeer, *et al.,* "Evaluating cryptographic vulnerabilities created by quantum computing in industrial control systems," *Homeland Security Operational Analysis Center,* RAND Corporation, 2023, doi: 10.7249/RRA2427-1.

[3] Cert Polska "Energy sector incident report – 29 December," 2026. [Online]. Available: https://cert.pl/en/posts/2026/01/incident-report-energy-sector-2025/

[4] Modbus Organisation, "Modbus/TCP Security protocol specification," 2021. [Online]. Available: https://www.modbus.org/file/secure/modbussecurityprotocol.pdf

[5] D. U. Group, "Overview of DNP3 Security Version 6," 2025. [Online]. Available: https://www.dnp.org/About/Overview-of-DNP3-Protocol

[6] Profibus, "Profinet Security," 2026. [Online]. Available: https://www.profinet.com/profinet-explained/security

[7] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science,* Santa Fe, NM, USA, 1994, pp. 124-134, doi: 10.1109/SFCS.1994.365700.

[8] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing (STOC '96),* ACM, Philadelphia, PA, USA, pp. 212–219, doi: 10.1145/237814.237866.

[9] International Electrotechnical Commission, "Security for industrial automation and control systems, Part 3-3: System security requirements and security levels," IEC Standard 62443-3-3, 2019.

[10] International Electrotechnical Commission, "Power systems management and associated information exchange - Data and communications security - Part 5: Security for IEC 60870-5 and derivatives," IEC Standard 62351-5, 2023.

[11] A. Castiglione, J. G. Esposito, V. Loia, M. Nappi, C. Pero, and M. Polsinelli, "Integrating post-quantum cryptography and blockchain to secure low-cost IoT devices," in *IEEE Transactions on Industrial Informatics,* 21(2), 2025, pp. 1674–1683, doi: 10.1109/TII.2024.3485796.

[12] S. Ghosh, M. Zaman, R. Joshi, and S. Sampalli, "Multi-phase quantum-resistant framework for secure communication in SCADA systems," in *IEEE Transactions on Dependable and Secure Computing,* 21(6), 2024, pp. 5461–5478, doi: 10.1109/TDSC.2024.3378474.

[13] G. Alagic, *et al.,* "Status report on the fourth round of the NIST post-quantum cryptography standardization process," *NIST Interagency/Internal Report (NISTIR)*, NISTIR 8545, National Institute of Standards and Technology, Gaithersburg, MD, USA 2025, doi: 10.6028/NIST.IR.8545.

[14] S. Paul, *et al.,* "Towards post-quantum security for cyber-physical systems: Integrating PQC into industrial M2M communication," *J. Comput. Secur.* 30(4), 2022, pp. 623–653, doi: 10.3233/JCS-210037

[15] National Institute of Standards and Technology, "Module-Lattice-Based Digital Signature Standard," NIST FIPS 204, 2024, doi: 10.6028/NIST.FIPS.203.

[16] National Institute of Standards and Technology, "Module-Lattice-Based Key-Encapsulation Mechanism," NIST FIPS 203, 2024, doi: 10.6028/NIST.FIPS.204.

[17] "Automated validation of internet security protocols and applications," 2003. [Online]. Available: http://www.avispa-project.org/

[18] Open Quantum Safe Project, "Liboqs," 2025. [Online]: Available: https:// https://openquantumsafe.org/

[19] S. Raimbault, "Libmodbus," 2025. [Online]: Available: https://libmodbus.org

[20] D. Pan, *et al.,* "The Evolution of Quantum Secure Direct Communication: On the Road to the Qinternet," *IEEE Communications Surveys & Tutorials,* 2024, doi: 10.48550/arXiv.2311.13974.

[21] "The Transport Layer Protocol," IETF RFC 5246 2008. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc5246

[22] OpenSSL, "OpenSSL Library," 2025, [Online]. Available: https://openssl-library.org/