ML-DSA Digital Signatures in Resource-Constrained MQTT Environments

Jiahao Xiang, Lang Li and Jingya Feng

***Abstract*—The emergence of large-scale quantum computers necessitates migration of Internet of Things (IoT) systems to post-quantum cryptographic standards. This work evaluates ML-DSA (Module-Lattice-Based Digital Signature Algorithm) integration within MQTT-based IoT systems through performance analysis on ARM Cortex-M4 microcontrollers. Performance measurements quantify 70–122× computational overhead relative to ECDSA, with signature generation latencies of 657–1,150 ms across parameter sets.**

***Index Terms*—Post-Quantum Cryptography, ML-DSA, MQTT Protocol, IoT Security, Resource-Constrained Devices**

I. Introduction

THE emergence of quantum computing fundamentally undermines current cryptographic infrastructures, necessitating migration to post-quantum standards [1]. The National Institute of Standards and Technology (NIST) has formalized ML-DSA (Module-Lattice-Based Digital Signature Algorithm) within FIPS 204 [2] as the primary standard for post-quantum digital signatures.

Post-quantum signature schemes impose substantial overhead compared to classical alternatives. ML-DSA signatures span 2,420–4,627 bytes across security levels, representing 30–70× size increases relative to 64-byte ECDSA signatures [3]. These expanded signatures, combined with elevated computational demands, substantially exceed the capabilities of resource-constrained devices [4].

Internet of Things (IoT) systems exemplify these deployment challenges. The MQTT protocol, widely adopted for IoT messaging due to its lightweight design, experiences performance degradation when post-quantum signatures introduce overhead on resource-constrained devices. Despite performance overhead, signature-based authentication remains essential for applications requiring cryptographic non-repudiation, audit trails, and public key infrastructure compatibility. This disparity between standardization progress and deployment feasibility motivates systematic performance characterization.

This research is supported by the Open Fund of Hunan Engineering Research Center for Cyberspace Security Technology and Application at Hengyang Normal University (2025HSKFJJ031), "the 14th Five-Year Plan" Key Disciplines and Application-oriented Special Disciplines of Hunan Province (Xiangjiaotong [2022] 351), the Science and Technology Innovation Program of Hunan Province (2016TP1020). *(Corresponding author: Lang Li.)*

The authors are with the Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, the Hunan Engineering Research Center of Cyberspace Security Technology and Applications, and the College of Computer Science and Technology, Hengyang Normal University, Hengyang 421002, China (e-mail: jiahaoxiang2000@gmail.com; lilang911@126.com; fengjyk@126.com).

This work addresses ML-DSA integration within MQTT-based IoT systems through performance analysis on ARM Cortex-M4 microcontrollers. Two contributions advance post-quantum IoT deployment:

1) **Performance Benchmarking**: Cycle-accurate measurements of ML-DSA signature operations on ARM Cortex-M4 microcontrollers at 168 MHz across all three standardized parameter sets, quantifying execution latency and throughput.
2) **MQTT Integration Assessment**: End-to-end latency and message size overhead evaluation within MQTT publish-subscribe workflows, comparing ML-DSA against ECDSA P-256 baseline.

The remainder of this paper is organized as follows: Section II presents background and related work. Section III describes the system architecture. Section IV presents experimental evaluation. Section V concludes with implications for deployment.

II. Background and Related Work

A. ML-DSA Algorithm

ML-DSA constitutes NIST's standardized post-quantum digital signature scheme (FIPS 204 [2]) based on CRYSTALS-Dilithium. The algorithm employs the Fiat-Shamir with Aborts paradigm over polynomial rings $R_q = \mathbb{Z}_q[X]/(X^{256}+1)$ with $q = 8380417$, deriving security from Module Learning With Errors (MLWE) and Module Short Integer Solution (MSIS) assumptions. Polynomial arithmetic utilizes Number Theoretic Transform (NTT) operations achieving $O(n \log n)$ complexity.

ML-DSA implements rejection sampling requiring iterative signature generation with expected iteration counts of 4.25, 5.1, and 3.85 for ML-DSA-44, ML-DSA-65, and ML-DSA-87 respectively, directly impacting timing predictability. NIST standardizes three parameter sets with distinct security-performance trade-offs. Table I summarizes implementation characteristics.

TABLE I: ML-DSA Parameter Sets

Parameter	ML-DSA-44	ML-DSA-65	ML-DSA-87
Security Category	2 (AES-128)	3 (AES-192)	5 (AES-256)
Matrix (k, ℓ)	$(4, 4)$	$(6, 5)$	$(8, 7)$
Private Key (bytes)	2,560	4,032	4,896
Public Key (bytes)	1,312	1,952	2,592
Signature (bytes)	2,420	3,309	4,627

B. MQTT Protocol

Message Queuing Telemetry Transport (MQTT) constitutes an OASIS standard messaging protocol implementing publish-subscribe architecture with minimal overhead for resource-constrained IoT devices. MQTT specifies three Quality of Service levels: QoS 0 (fire-and-forget), QoS 1 (guaranteed delivery via PUBACK), and QoS 2 (exactly-once delivery). Native security provisions include username/password authentication and TLS integration, but lack built-in digital signature support.

ML-DSA integration within MQTT environments presents implementation challenges. Signatures span 2,420–4,627 bytes compared to 64 bytes for ECDSA, representing 38×–72× overhead. For typical IoT sensor data (10–100 bytes), signatures dominate packet composition. ARM Cortex-M4 platforms require hundreds of milliseconds for signature generation, creating processing bottlenecks that may violate MQTT responsiveness guarantees.

C. Related Work

Banegas et al. [5] benchmarked CRYSTALS-Dilithium on embedded systems, reporting computational overhead relative to ECDSA on ARM Cortex-M4 processors. The pqm4 benchmarking campaign [6] extends these observations to standardized ML-DSA parameter sets, reporting tens of kilobytes memory consumption and millions of CPU cycles per signature on Cortex-M4 targets.

Practical deployment scenarios reveal performance bottlenecks. Analysis of the SUIT (Software Update for the Internet of Things) framework demonstrates post-quantum signature verification requiring up to 3.2 seconds on low-power microcontrollers, exceeding sub-second latency constraints for real-time applications. Marchsreiter [7] reports order-of-magnitude transaction throughput reductions on embedded blockchain nodes with ML-DSA. Fault injection research [8] achieved 89.5% attack success rates on ARM Cortex-M ML-DSA implementations through electromagnetic fault injection.

Kim and Seo [9] demonstrate that post-quantum signatures introduce prohibitive MQTT authentication overhead, motivating KEM-based architectures. Their CRYSTALS-Kyber implementation achieves 4.32-second handshake completion on 8-bit AVR microcontrollers. Barrett multiplication techniques achieve 1.38–1.51× performance improvements on ARM Cortex-M3 [10]. However, empirical studies evaluating ML-DSA performance within full MQTT protocol implementations remain limited, representing a knowledge gap in post-quantum IoT deployment.

Recent MQTT-specific evaluations further characterize migration costs. Samandari and Gritti [11] analyze post-quantum authentication in MQTT using Dilithium and report latency and payload expansion relative to classical baselines. Rampazzo and Henriques [12] quantify hybrid post-quantum overhead in MQTT, showing that hybrid handshakes and signature bundles increase end-to-end latency and message sizes even in short publish-subscribe exchanges.

Major transfer issues for post-quantum migration are increasingly framed as crypto-agility and deployment coordination problems. Large-scale traffic inspection indicates substantial lag in quantum-readiness and emphasizes hybrid key exchange, certificate lifecycle adjustments, and transition monitoring as prerequisites for migration [13]. Performance studies in TLS and SSH further show that handshake latency and certificate size overheads remain the dominant integration barriers, with operational trade-offs between interoperability and security assurances [14], [15]. These observations align with IoT constraints where bandwidth and energy budgets magnify protocol-level overhead.

III. System Architecture and Adaptive Protocol

A. Hardware Platform

The experimental platform employs STM32F407VG development boards featuring ARM Cortex-M4F cores with hardware floating-point unit operating at 168 MHz. Memory resources comprise 1 MB Flash memory for program storage and 192 KB SRAM for runtime operations. Network connectivity is provided through ESP32-WROOM-32 wireless modules interfaced via UART at 115,200 baud, implementing IEEE 802.11n WiFi connectivity with integrated TCP/IP stack. Energy measurement employs INA219 current sensor modules at 12-bit resolution with ±0.8 mA precision.

B. Software Architecture

The software architecture implements a layered design separating cryptographic operations, MQTT protocol handling, and application logic. The cryptographic layer employs the pqm4 reference library [6] optimized for ARM Cortex-M4, providing all three ML-DSA parameter sets with verified NIST test vector compliance. The library exports three primary API functions: `crypto_sign_keypair()` for key generation, `crypto_sign()` for signature generation, and `crypto_sign_verify()` for validation.

The MQTT protocol layer utilizes the Eclipse Paho MQTT Embedded C client configured for QoS 1 operation with 5-second keepalive intervals, communicating with a Mosquitto MQTT broker. Configuration parameters include 256-byte receive buffers and 5,120-byte transmit buffers supporting ML-DSA-87 signatures plus application data.

C. Signature Integration

Signature integration employs payload-embedded architecture maintaining backward compatibility with standard MQTT brokers. The composite message format implements type-length-value (TLV) encoding: 1-byte message type identifier, 2-byte payload length, variable-length application payload, 1-byte signature algorithm identifier, 2-byte key identifier, 4-byte Unix timestamp for replay attack mitigation, 2-byte signature length, and variable-length ML-DSA signature data (2,420–4,627 bytes).

This format enables subscribers to parse messages without prior knowledge of signature algorithms through self-describing metadata fields. The TLV structure accommodates future cryptographic algorithm upgrades through algorithm identifier extension without protocol-level modifications.

D. Cryptographic Optimization

The implementation incorporates optimization techniques targeting NTT operations and modular arithmetic. NTT operations dominate computational cost, consuming 60–70% of total signing cycles. Key optimizations include:

Assembly Optimization: Hand-optimized ARM assembly exploits instruction-level parallelism and efficient use of the `UMULL` instruction for 32×32→64-bit multiplication.

Montgomery Reduction: Replacing division-based modular reduction with multiplication-based techniques eliminates expensive division operations, achieving 25–35% reduction overhead improvement.

Lazy Reduction: Deferring modular reduction across multiple butterfly operations reduces reduction frequency, achieving 15–25% NTT latency improvement.

Combined optimization techniques achieve 40–50% performance improvement relative to reference implementations.

IV. Experimental Evaluation

A. Methodology

1) Measurement Framework: Performance profiling exploits ARM Cortex-M4 Data Watchpoint and Trace (DWT) hardware for cycle-accurate measurement through the `DWT_CYCCNT` register. This hardware approach eliminates software profiling overhead and achieves single-cycle temporal resolution. Measurement accuracy was validated through comparison with external logic analyzer traces, confirming ± 1 cycle precision.

Memory utilization analysis combines static and dynamic measurement techniques. Static memory consumption is quantified via `arm-none-eabi-size` toolchain utilities. Dynamic memory profiling utilizes stack watermarking techniques with 0xDEADBEEF sentinel values at 32-byte intervals. Energy consumption assessment employs INA219 current sensor modules measuring supply current at 100 Hz sampling frequency.

2) Software Configuration: The software environment employs ARM GCC toolchain version 10.3.1 with `-O3` optimization. Preliminary testing revealed `-O3` achieved 18–23% performance improvement over `-O2` with 12–15% code size increase. ML-DSA implementations employ the pqm4 reference library with verified NIST test vector compliance. ECDSA baseline measurements employ the micro-ecc library with NIST P-256 curves.

MQTT protocol integration utilizes Eclipse Paho MQTT Embedded C client communicating with Mosquitto MQTT broker (version 2.0.15). Network infrastructure employs IEEE 802.11n (2.4 GHz band) with controlled signal strength at -45 to -52 dBm and $<0.1\%$ packet loss.

3) Benchmark Design: Each parameter set undergoes evaluation across representative IoT message payloads: 10-byte (single-sensor readings, 23% of observed traffic), 50-byte (multi-parameter telemetry, 51%), and 100-byte (diagnostic reports, 18%). Statistical rigor is ensured through 1,000-iteration repeated measurements with IQR-based outlier elimination (rejection rates of 0.8–2.3%). All measurements were conducted under temperature-controlled conditions (25°C $\pm$2°C) with device voltage stabilized at 3.3V $\pm$1%.

B. Results and Analysis

1) Computational Performance: Computational performance measurements employ the pqm4 library implementation incorporating optimization techniques described in Section III. All reported cycle counts reflect optimized implementations achieving 40–50% performance improvement relative to reference implementations.

2) Key Generation: Table II presents key generation performance across ML-DSA parameter sets.

TABLE II: Key Generation Performance (168 MHz)

Scheme	Cycles	Time	Ops/s	Overhead
ECDSA P-256	252,000	1.50 ms	666.7	1.00×
ML-DSA-44	25,368,000	151 ms	6.6	100.7×
ML-DSA-65	41,832,000	249 ms	4.0	166.0×
ML-DSA-87	59,976,000	357 ms	2.8	238.0×

The 100.7–238× computational overhead reflects lattice-based cryptographic complexity. Measured execution times of 151–357 ms establish feasibility for infrequent key generation during device provisioning but prohibit high-frequency rotation strategies.

3) Signature Generation: Signature generation represents the primary performance bottleneck. Table III quantifies signing performance.

TABLE III: Signature Generation Performance (168 MHz)

Scheme	Payload	Time	Ops/s	Overhead
ECDSA P-256	10 B	9.19 ms	108.8	1.00×
ECDSA P-256	50 B	9.39 ms	106.5	1.00×
ECDSA P-256	100 B	9.59 ms	104.3	1.00×
ML-DSA-44	50 B	663 ms	1.51	70.6×
ML-DSA-65	50 B	853 ms	1.17	90.8×
ML-DSA-87	50 B	1,136 ms	0.88	121.0×

Signing latencies of 657–1,150 ms across parameter sets remain 70–122× slower than ECDSA despite optimization. Performance variability from rejection sampling introduces timing non-determinism requiring worst-case latency analysis for real-time applications.

4) Signature Verification: Table IV presents verification latency measurements.

TABLE IV: Signature Verification Performance (168 MHz)

Scheme	Payload	Time	Ops/s	Overhead
ECDSA P-256	50 B	16.2 ms	61.7	1.00×
ML-DSA-44	50 B	420 ms	2.38	25.9×
ML-DSA-65	50 B	533 ms	1.88	32.9×
ML-DSA-87	50 B	710 ms	1.41	43.8×

Verification executes deterministically without rejection sampling, yielding predictable latency characteristics. Verification operations achieve 26–44× overhead relative to ECDSA compared to 70–122× for signature generation.

5) Memory Utilization: Table V presents static and dynamic memory requirements.

Memory requirements of 22.7–43.1 KB SRAM constrain deployment to mid-range microcontrollers. Flash requirements

TABLE V: Memory Requirements

Scheme	Flash	Stack	Keys	Total SRAM
ECDSA P-256	9.7 KB	0.8 KB	0.1 KB	2.1 KB
ML-DSA-44	37.2 KB	6.4 KB	3.8 KB	22.7 KB
ML-DSA-65	54.8 KB	8.7 KB	5.8 KB	32.8 KB
ML-DSA-87	73.9 KB	11.2 KB	7.3 KB	43.1 KB

of 37.2–73.9 KB establish deployment feasibility for microcontrollers with limited Flash capacity in cost-constrained IoT applications.

6) Protocol-Level Overhead:

a) Message Size: Table VI quantifies message size overhead for signed MQTT payloads.

TABLE VI: MQTT Message Size Overhead

Scheme	Payload	Signed	Unsigned	Ratio
ECDSA P-256	50 B	122 B	58 B	2.1×
ML-DSA-44	50 B	2,478 B	58 B	42.7×
ML-DSA-65	50 B	3,367 B	58 B	58.1×
ML-DSA-87	50 B	4,685 B	58 B	80.8×

Message size overhead directly impacts network bandwidth consumption and transmission costs in cellular IoT deployments where data transfer incurs per-byte charges.

b) End-to-End Latency: Table VII presents complete publish-subscribe workflow latency incorporating signature generation, network transmission, and verification.

TABLE VII: End-to-End MQTT Latency

Scheme	Sign	Net.	Verify	Total	Overhead
ECDSA P-256	9.4 ms	28.5 ms	16.2 ms	54.1 ms	1.00×
ML-DSA-44	663 ms	31.2 ms	420 ms	1,114 ms	20.6×
ML-DSA-65	853 ms	33.8 ms	533 ms	1,420 ms	26.2×
ML-DSA-87	1,136 ms	37.4 ms	710 ms	1,883 ms	34.8×

End-to-end latency of 1.11–1.88 seconds exceeds sub-second bounds for interactive IoT applications, necessitating architectural accommodations for latency-sensitive deployments.

c) Sustainable Throughput: Sustainable message rates range from 0.87 to 1.52 messages per second across ML-DSA parameter sets, compared to 104–109 messages per second for ECDSA. Signature generation constitutes the primary performance bottleneck, representing throughput degradation of 70–122× relative to classical signature schemes.

7) Trade-off Analysis: Table VIII quantifies security-performance trade-offs normalized to ML-DSA-44.

TABLE VIII: ML-DSA Parameter Set Trade-offs (Normalized to ML-DSA-44)

Metric	DSA-44	DSA-65	DSA-87
Security Level	2 (AES-128)	3 (AES-192)	5 (AES-256)
Signing Time	1.00×	1.30×	1.75×
Verification Time	1.00×	1.27×	1.72×
Signature Size	1.00×	1.37×	1.91×
Total SRAM	1.00×	1.45×	1.90×

For IoT deployments with 5–10 year operational lifetimes under current quantum computing development trajectories, NIST security level 2 (ML-DSA-44) provides adequate quantum resistance with minimal resource overhead. Long-term infrastructure deployments (20+ years) requiring conservative security margins justify ML-DSA-87 despite 75–90% resource overhead increases.

8) Deployment Feasibility: Deployment feasibility analysis evaluates ML-DSA applicability across representative IoT application categories based on measured performance characteristics.

High-throughput sensor networks requiring frequent authenticated message publication encounter computational bottlenecks limiting sustainable publication rates to 0.87–1.52 messages per second across ML-DSA parameter sets, compared to 104–109 messages per second for ECDSA. Deployments requiring publication frequencies exceeding 1 message per second necessitate architectural mitigation strategies: message aggregation combining multiple sensor readings into single signed payloads (amortizing signature overhead across N measurements), publisher-side caching reducing redundant signing of identical state values, or hybrid authentication employing ML-DSA signatures for security-critical messages with MAC-based authentication for routine telemetry. For networks with 100 sensor nodes each publishing at 0.01 Hz, aggregate network throughput of 1 message/second remains within ML-DSA-44 computational capacity.

Battery-powered devices operating under energy constraints require power consumption analysis. Signature generation computational cost of 110.4 million cycles at 168 MHz consuming 657 milliseconds implies power draw of approximately 50 mW (assuming 3.3V supply at 15 mA typical active current). Energy per signature operation totals 32.9 mJ for ML-DSA-44. For devices transmitting 100 authenticated messages daily from 2,000 mAh batteries (3.3V nominal voltage, 23.8 kJ total energy), ML-DSA signature generation consumes 3.29 J daily (13.8% of total energy budget). This analysis excludes network transmission energy and verification overhead at subscriber devices. ML-DSA remains viable for battery-powered publishers with daily-scale message frequencies; higher publication rates require energy harvesting or mains power.

Real-time control systems with sub-second latency requirements encounter timing constraint violations from ML-DSA end-to-end authentication latency of 1.11–1.88 seconds, exceeding sub-second responsiveness bounds for interactive control applications. Deployments requiring real-time authenticated command-response interactions necessitate architectural accommodations: asymmetric publisher-subscriber security models where publishers employ fast signing with subscribers performing offline ML-DSA verification for audit trail generation, pre-signed command templates enabling instant transmission of pre-authenticated control messages with restricted command spaces, or hybrid cryptographic modes utilizing classical signatures for time-critical operations with periodic ML-DSA re-authentication for long-term security. Pure ML-DSA authentication remains suitable for monitoring applications tolerating multi-second latency but inappropriate for closed-loop control systems requiring deterministic sub-second response.

9) Optimization Opportunities: Several optimization strategies mitigate ML-DSA performance overhead in resource-constrained deployments.

Parameter set selection: Deployment-specific security requirement analysis enables ML-DSA-44 selection, reducing computational overhead by 42.2% (signing latency: 1,150 ms $\rightarrow$ 657 ms), signature size overhead by 47.7% (4,627 bytes $\rightarrow$ 2,420 bytes), and memory consumption by 47.3% (43.1 KB $\rightarrow$ 22.7 KB) relative to ML-DSA-87 while maintaining NIST security level 2 quantum resistance.

Message aggregation: Batch signature generation over aggregated sensor readings amortizes cryptographic overhead across multiple measurements. Aggregating N measurements into a single signed payload reduces per-measurement signing overhead from 657 ms to $657/N$ ms at cost of $N\times$ sampling-interval latency increase.

Hybrid authentication: Selective ML-DSA deployment for security-critical messages combined with MAC-based authentication for routine telemetry reduces average authentication overhead while maintaining non-repudiation for critical operations. For deployment patterns comprising 90% routine telemetry (MAC: 0.5 ms) and 10% critical messages (ML-DSA-44: 657 ms), weighted average overhead totals 66.2 ms compared to 657 ms for pure ML-DSA deployment.

Hardware acceleration: Future ARM Cortex-M processors incorporating cryptographic acceleration extensions for NTT operations could provide 5–10$\times$ performance improvements, reducing ML-DSA signing latency to sub-100-millisecond ranges.

V. Conclusion

This work presents performance characterization of ML-DSA integration within MQTT-based IoT systems on ARM Cortex-M4 microcontrollers. Signature generation latencies of 657–1,150 ms represent 70–122$\times$ overhead compared to ECDSA P-256, establishing signature generation as the primary computational bottleneck limiting sustainable message throughput to 0.87–1.52 messages per second. Memory requirements of 22.7–43.1 KB SRAM and 37.2–73.9 KB Flash constrain deployment to mid-range microcontrollers.

ML-DSA remains viable for applications tolerating multi-second latency and sub-hertz publication frequencies, including environmental monitoring, asset tracking, and periodic telemetry reporting. Real-time control systems requiring sub-second response necessitate architectural accommodations or hybrid authentication approaches. Future research directions include hardware acceleration evaluation on emerging ARM Cortex-M processors with cryptographic extensions and batch signature verification schemes for gateway devices.

References

[1] A. Khalid, S. McCarthy, W. Liu, and M. O'Neill, "Lattice-based cryptography for IoT in a quantum world: Are we ready?" Cryptology ePrint Archive, Paper 2019/681, 2019.

[2] NIST, "Fips 204: Module-lattice-based digital signature standard," Federal Information Processing Standards Publication, 2024, available at: https://csrc.nist.gov/pubs/fips/204/final.

[3] V. Hwang, Y. Kim, and S. C. Seo, "Multiplying polynomials without powerful multiplication instructions (long paper)," Cryptology ePrint Archive, Paper 2024/1649, 2024.

[4] S. Ghosh, R. Misoczki, and M. R. Sastry, "Lightweight post-quantum-secure digital signature approach for IoT motes," Cryptology ePrint Archive, Paper 2019/122, 2019.

[5] G. Banegas, K. Zandberg, E. Baccelli, A. Herrmann, and B. Smith, "Quantum-resistant security for software updates on low-power networked embedded devices," in *Applied Cryptography and Network Security*, ser. ACNS 2022. Springer, 2022, also available at: https://eprint.iacr.org/2021/781.

[6] M. J. Kannwischer, M. Krausz, R. Petri, and S.-Y. Yang, "pqm4: Benchmarking NIST additional post-quantum signature schemes on microcontrollers," Cryptology ePrint Archive, Paper 2024/112, 2024.

[7] D. Marchsreiter, "Towards quantum-safe blockchain: Exploration of pqc and public-key recovery on embedded systems," Cryptology ePrint Archive, Paper 2024/1178, 2024.

[8] Y. Wang, J. Yu, S. Qu, X. Zhang, X. Li, C. Zhang, and D. Gu, "Mind the faulty keccak: A practical fault injection attack scheme apply to all phases of ml-kem and ml-dsa," Cryptology ePrint Archive, Paper 2024/1522, 2024.

[9] Y. Kim and S. C. Seo, "An optimized instantiation of post-quantum MQTT protocol on 8-bit AVR sensor nodes," in *Proceedings of the 2025 ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '25. ACM, 2025, pp. 1–19.

[10] V. Hwang, Y. Kim, and S. C. Seo, "Multiplying polynomials without powerful multiplication instructions," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2025, no. 1, pp. 160–202, 2025, extended version of Barrett Multiplication for Dilithium on Embedded Devices.

[11] J. Samandari and C. Gritti, "Post-quantum authentication in the MQTT protocol," *Journal of Cybersecurity and Privacy*, vol. 3, no. 3, pp. 416–434, 2023.

[12] F. J. A. Rampazzo and M. A. A. Henriques, "Assessment of the impact of hybrid post-quantum cryptography on the performance of the MQTT communication protocol," in *SIoT 2023 - IEEE International Conference on Smart Internet of Things*. IEEE, 2023, pp. 1–5.

[13] S. Cho, Y. Hyoung, H. Kim, M. Sim, A. Chattopadhyay, H. Seo, and H. Kim, "Toward crypto agility: Automated analysis of quantum-vulnerable TLS via packet inspection," Cryptology ePrint Archive, Paper 2025/1549, 2025.

[14] P. Kampanakis, D. Sikeridis, and M. Devetsikiotis, "Post-quantum authentication in tls 1.3: A performance study," in *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society, 2020.

[15] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis, "Assessing the overhead of post-quantum cryptography in tls 1.3 and ssh," in *Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies*. ACM, 2020, pp. 149–156.

Jiahao Xiang is pursuing a Master's degree in Electronic Information at Hengyang Normal University, China. His research focuses on cryptographic engineering and efficient implementations of block ciphers on resource-constrained devices. Publications include works on lightweight cryptography optimization and contributions to open-source cryptographic projects.

Lang Li received his Ph.D. and Master's degrees in computer science from Hunan University, Changsha, China, in 2010 and 2006, respectively, and earned his B.S. degree in circuits and systems from Hunan Normal University in 1996. Since 2011, he has been working as a professor in the College of Computer Science and Technology at the Hengyang Normal University, Hengyang, China. He has research interests in embedded system and information security.

Jingya Feng received the M.S. degree from Hunan Normal University, Changsha, China, in 2021, and the Ph.D. degree from Guilin University of Electronic Science and Technology, Guilin, China, in 2025. She Joined the School of Computer Science and Technology, Hengyang Normal University in July 2025. Her main research interests include cryptology, with particular focus on the design and optimized implementation of symmetric encryption schemes.