IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN XXXX-XXXX, Vol. 0, No. 0, pp. 1–14. DOI:XXXXXXXX

ML-DSA Digital Signatures in Resource-Constrained MQTT Environments

Jiahao Xiang[1] and Lang Li[1]

Hengyang Normal University, College of Computer Science and Technology, Hengyang, China

Abstract. The imminent threat of large-scale quantum computers necessitates the migration of Internet of Things (IoT) systems to post-quantum cryptographic standards. While NIST has standardized ML-DSA (Module-Lattice-Based Digital Signature Algorithm) for digital signatures, the practical deployment of post-quantum authentication in resource-constrained IoT environments remains unexplored. This research evaluates ML-DSA integration within MQTT-based IoT systems through comprehensive performance analysis on ARM Cortex-M4 microcontrollers. The methodology encompasses signature generation and verification benchmarking, memory utilization analysis, and protocol overhead assessment under realistic IoT constraints. Analysis examines the performance implications of ML-DSA deployment compared to classical signature schemes, quantifying computational overhead, memory requirements, and verification latency on resource-constrained devices. These findings reveal fundamental trade-offs between post-quantum security and IoT performance requirements, providing critical insights for practical deployment strategies in resource-limited environments.

Keywords: Post-Quantum Cryptography · ML-DSA · MQTT Protocol · IoT Security · Resource-Constrained Devices

1 Introduction

The emergence of quantum computing poses an existential threat to current cryptographic infrastructures, necessitating systematic migration to post-quantum cryptographic standards across all computing domains [KMLO19]. The National Institute of Standards and Technology (NIST) has formalized ML-DSA (Module-Lattice-Based Digital Signature Algorithm) within FIPS 204 [NIS24], establishing this CRYSTALS-Dilithium-based scheme as the primary standard for post-quantum digital signatures.

The transition from theoretical post-quantum standardization to practical deployment has revealed fundamental implementation challenges that extend beyond algorithmic considerations [Ano24]. Post-quantum signature schemes impose substantial computational and storage overhead compared to classical alternatives. ML-DSA signatures range from 2,420 bytes to 4,627 bytes across security levels, representing 30-70$\times$ size increases relative to 64-byte ECDSA signatures. These expanded signature sizes, combined with elevated computational demands, frequently exceed the processing capabilities of resource-constrained devices [HKS24].

Internet of Things (IoT) systems exemplify these deployment challenges, where computational, memory, and energy limitations fundamentally constrain cryptographic implementation choices [GMS19]. Despite performance overhead, signature-based authentication mechanisms remain essential for applications requiring cryptographic non-repudiation, comprehensive audit trails, and compatibility with existing public key infrastructure frameworks. The MQTT protocol, widely adopted for IoT messaging due to its lightweight characteristics, becomes particularly problematic when post-quantum signatures impose prohibitive performance overhead on resource-constrained devices. This disparity creates a critical gap between standardization achievements and practical deployment feasibility in IoT environments.

This research systematically addresses the challenge of ML-DSA integration within MQTT-based IoT systems through comprehensive performance analysis on ARM Cortex-M4 microcontrollers. The work provides three primary contributions to post-quantum IoT deployment:

- **Computational Performance Benchmarking**: Comprehensive benchmarking of ML-DSA signature operations on resource-constrained ARM Cortex-M4 microcontrollers, quantifying computational overhead, execution latency, and performance variations across all three standardized ML-DSA parameter sets under realistic IoT operating conditions.
- **Memory Utilization Analysis**: Systematic memory utilization analysis for ML-DSA deployment in constrained environments, measuring static storage requirements, dynamic memory allocation patterns, and peak memory consumption during signature generation and verification operations.
- **Protocol-Level MQTT Integration Assessment**: Protocol-level overhead evaluation of ML-DSA integration within MQTT communication frameworks, analyzing message size increases, transmission latency impacts, and network throughput degradation relative to classical signature schemes.

Through comparative analysis with classical signature schemes, these contributions establish fundamental trade-offs between post-quantum security guarantees and IoT performance requirements, providing essential insights for practical deployment strategies in resource-limited environments.

The remainder of this paper is organized as follows: Section 2 presents background information on post-quantum cryptography and related work in IoT deployments. Section 3 provides an overview of the ML-DSA algorithm and its implementation considerations. Section 4 describes the implementation architecture for MQTT-based IoT systems. Section 5 details the experimental methodology for performance evaluation on ARM Cortex-M4 microcontrollers. Section 6 presents and analyzes experimental results, examining the trade-offs between security and performance. Finally, Section 7 concludes with implications for practical deployment and future research directions.

2 Related Work and Motivation

While conventional network protocols have undergone extensive analysis for post-quantum migration [KSD20, SKD20], IoT-specific communication protocols remain inadequately addressed, creating deployment barriers in resource-constrained environments.

2.1 ML-DSA Performance Benchmarks on Embedded Systems

Banegas et al. [BZB+22] quantified these performance implications through comprehensive benchmarking on embedded systems, establishing that CRYSTALS-Dilithium signature operations require approximately 45% additional computational cycles compared to classical ECDSA implementations on ARM Cortex-M4 processors. This computational overhead compounds with memory constraints to create deployment bottlenecks in IoT environments.

The ongoing pqm4 benchmarking campaign extends these observations to the standardized ML-DSA parameter sets, reporting that even aggressively optimized implementations still consume tens of kilobytes of static and dynamic memory and millions of CPU cycles per signature on Cortex-M4 targets [KKPY24]. Complementary measurements on higher performance IoT-class microcontrollers demonstrate that migrating to newer cores such as Cortex-M7 reduces latency but leaves signature generation and verification firmly in the tens-of-milliseconds regime, keeping ML-DSA near the edge of acceptable responsiveness for interactive device workloads [HW23].

2.2 Deployment Bottlenecks in IoT Applications

Practical deployment scenarios reveal critical performance bottlenecks that challenge IoT system viability. Analysis of the SUIT (Software Update for the Internet of Things) framework demonstrates that post-quantum signature verification operations require up to 3.2 seconds on low-power microcontrollers, substantially exceeding acceptable latency constraints for real-time IoT applications.

These performance constraints are compounded by security vulnerabilities and protocol-level throughput ceilings. Marchsreiter [Mar24] shows that blockchain workloads on embedded nodes experience order-of-magnitude drops in transactions-per-second once ML-DSA is introduced, with signing latency alone dominating system throughput. Fault injection research targeting ML-DSA and ML-KEM implementations achieved 89.5% attack success rates on ARM Cortex-M processors through electromagnetic fault injection techniques [WYQ+24]. The analyses demonstrate that Keccak-based hash functions—integral to ML-DSA randomness generation and signature computation—exhibit particular susceptibility to loop-abort faults enabling complete private key recovery, necessitating additional countermeasures that further impact performance.

2.3 Alternative Approaches and Limitations

Recent research has explored alternative authentication architectures to address these deployment challenges. Kim and Seo [KS25] demonstrate that direct application of post-quantum signatures to MQTT authentication introduces prohibitive performance overhead, prompting KEM-based authentication architectures that eliminate signature operations entirely. While their CRYSTALS-Kyber implementation achieves 4.32-second handshake completion on 8-bit AVR microcontrollers, this approach circumvents rather than resolves the fundamental challenge of post-quantum signature deployment in signature-dependent applications.

Current algorithmic optimization research reveals limitations in addressing fundamental resource constraints. Barrett multiplication techniques achieve 1.38-1.51× performance improvements on ARM Cortex-M3 processors and 6.37-7.27× improvements on 8-bit AVR platforms [HKS25]. However, these optimizations provide insufficient performance gains to bridge the gap between post-quantum signature requirements and IoT device capabilities, necessitating comprehensive system-level analysis.

2.4 Research Gap and Motivation

The absence of comprehensive empirical studies specifically evaluating ML-DSA performance within MQTT protocol implementations represents a critical knowledge gap in post-quantum IoT deployment. This gap becomes particularly significant given MQTT's widespread adoption in industrial IoT deployments, where signature-based authentication remains mandatory for regulatory compliance and security audit requirements.

Existing research primarily focuses on isolated cryptographic operations or alternative protocol architectures, failing to address the systematic integration challenges inherent in MQTT-based IoT systems. This research addresses these limitations through comprehensive performance analysis of ML-DSA deployment within realistic MQTT environments, providing essential insights for practical post-quantum IoT migration strategies.

3 ML-DSA and MQTT Protocol Integration

3.1 ML-DSA Algorithm Characteristics

ML-DSA constitutes NIST's standardized post-quantum digital signature scheme (FIPS 204 [NIS24]) 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 security derived 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 across parameter sets, directly impacting timing predictability. The algorithm comprises three operations: key generation with $O(k \cdot \ell \cdot n \log n)$ complexity producing keys of 1.3-4.9 KB, computationally intensive signature generation with variable execution time, and deterministic verification with $O((k + \ell) \cdot n \log n)$ complexity—all substantially exceeding classical signature costs.

3.2 Parameter Sets and Security Analysis

NIST standardizes three ML-DSA parameter sets with distinct security-performance trade-offs. Table 1 summarizes the key parameters and resulting implementation characteristics.

Table 1: ML-DSA Parameter Sets and Implementation Characteristics

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
Expected Iterations	4.25	5.1	3.85

These parameter selections demonstrate the fundamental trade-off between quantum security strength and implementation overhead. Signature sizes increase by factors of 30-70$\times$ relative to classical ECDSA schemes, reflecting the inherent cost of lattice-based post-quantum security.

3.3 MQTT Protocol and Security Integration

Message Queuing Telemetry Transport (MQTT) constitutes an OASIS standard messaging protocol implementing publish-subscribe architecture with minimal overhead for resource-constrained IoT devices. MQTT utilizes binary packet structure comprising fixed header (2-byte mandatory component specifying packet type and control flags), variable header (packet-specific control information), and payload (message content up to 256 MB).

MQTT specifies three Quality of Service levels affecting signature integration: QoS 0 (fire-and-forget transmission), QoS 1 (guaranteed delivery via PUBACK acknowledgments), and QoS 2 (exactly-once delivery through four-way handshake). Native security provisions include username/password authentication, TLS integration, and X.509 certificate-based mutual authentication, but lack built-in digital signature support.

3.4 ML-DSA Integration Challenges and Performance Impact

ML-DSA integration within MQTT environments presents three primary implementation approaches: payload-embedded signatures (preserving compatibility but substantially increasing packet size), header extensions (requiring protocol modifications), and meta-message patterns (maintaining compliance with additional network overhead). Applications requiring non-repudiation necessitate asymmetric signatures like ML-DSA despite computational costs, while message authentication can utilize faster MACs with shared secrets.

ML-DSA signatures span 2,420-4,627 bytes compared to 64 bytes for ECDSA, representing $38\times$-$72\times$ overhead amplification. For typical IoT sensor data (10-100 bytes), signatures dominate packet composition, transforming 20-byte temperature measurements into 2.4-4.6 KB transmissions. ARM Cortex-M4 platforms require tens of milliseconds for signature generation and substantial memory resources (1.3-4.9 KB keys, tens of kilobytes working memory), creating processing bottlenecks that violate MQTT responsiveness guarantees.

Integration challenges include backward compatibility constraints, broker computational requirements, error handling extensions, processing power limitations, energy consumption escalation, and real-time constraint violations. These factors necessitate comprehensive empirical analysis to quantify performance trade-offs and inform practical deployment strategies within resource-constrained MQTT environments.

4 Implementation Architecture

5 Experimental Methodology

This section presents the experimental framework for evaluating ML-DSA integration within MQTT-based IoT systems. The methodology encompasses comprehensive performance benchmarking on resource-constrained ARM Cortex-M4 microcontrollers, systematic memory utilization analysis, and protocol-level overhead assessment under realistic IoT operational constraints.

5.1 Experimental Platform

The experimental platform targets ARM Cortex-M4 microcontrollers, representative of mid-range IoT devices deployed in industrial monitoring, smart agriculture, and building automation systems. The evaluation hardware comprises STM32F407VG development boards featuring ARM Cortex-M4F cores operating at 168 MHz with hardware floating-point unit, 1 MB Flash memory for program storage, and 192 KB SRAM for runtime operations. This configuration represents typical resource constraints encountered in contemporary IoT deployments requiring cryptographic authentication capabilities.

The software environment utilizes the ARM GCC toolchain (version 10.3.1) with `-O3` optimization enabling aggressive performance optimizations while maintaining standards compliance. ML-DSA implementation derives from the pqm4 reference library [KKPY24], providing optimized ARM Cortex-M4 implementations of all three standardized parameter sets (ML-DSA-44, ML-DSA-65, ML-DSA-87) with verified NIST test vector compliance. For comparative baseline measurements, the micro-ecc library implementing ECDSA with NIST P-256 curves is employed, compiled with identical optimization settings to ensure fair performance comparison.

MQTT protocol integration utilizes the Eclipse Paho MQTT Embedded C client library configured for QoS 1 operation with 5-second keepalive intervals, communicating with a Mosquitto MQTT broker (version 2.0.15) deployed on dedicated server infrastructure. Network connectivity employs IEEE 802.11n WiFi through ESP32-WROOM-32 wireless modules, eliminating physical wiring constraints while maintaining realistic IoT communication patterns.

5.2 Performance Measurement Framework

Computational performance measurement exploits ARM Cortex-M4 Data Watchpoint and Trace (DWT) hardware unit, providing cycle-accurate execution time quantification through the `DWT_CYCCNT` register. This hardware-based approach eliminates measurement overhead associated with software profiling while achieving single-cycle temporal resolution. Measurements capture complete operation execution including all preprocessing, core algorithm computation, and result formatting stages.

Memory utilization analysis employs dual-methodology assessment combining static and dynamic measurement techniques. Static memory consumption derives from compilation output through `arm-none-eabi-size` toolchain utilities, quantifying Flash memory requirements for ML-DSA implementation code and initialized data segments. Dynamic memory profiling utilizes stack watermarking techniques, initializing stack memory regions with distinctive patterns (0xDEADBEEF sentinel values) prior to operation execution and subsequently scanning for pattern corruption to determine peak stack utilization. Heap allocation monitoring instruments `malloc` and `free` functions to track runtime memory requests, though ML-DSA implementations typically avoid dynamic allocation in embedded contexts.

Energy consumption assessment, while not primary focus of this research, employs INA219 current sensor modules measuring supply current at 100 Hz sampling frequency throughout cryptographic operation execution. Voltage monitoring and current integration provide operation-level energy consumption estimates, enabling power-constrained deployment analysis for battery-operated IoT devices.

5.3 Benchmark Design

Benchmark methodology encompasses systematic evaluation of all ML-DSA cryptographic operations across standardized parameter sets. Key generation benchmarking measures complete public-private keypair generation including random seed generation, matrix expansion, and polynomial sampling operations. Signature generation benchmarking captures end-to-end signing latency including message hashing, rejection sampling iterations, and signature encoding, with statistical analysis accounting for variable iteration counts. Verification benchmarking measures signature validation computational cost including signature decoding, polynomial reconstruction, and validity checking operations.

Each parameter set (ML-DSA-44, ML-DSA-65, ML-DSA-87) undergoes evaluation across representative IoT message payloads spanning 10-byte minimal sensor readings (single temperature value), 50-byte typical telemetry packets (multi-sensor aggregated data), and 100-byte extended status reports (device diagnostics with metadata). This payload diversity captures realistic IoT communication patterns while enabling overhead ratio analysis across message sizes.

Baseline performance comparison employs ECDSA P-256 implementations executing identical operation sequences: keypair generation producing 32-byte private keys and 64-byte public keys, signature generation over equivalent message lengths producing 64-byte DER-encoded signatures, and signature verification operations. All measurements execute under identical environmental conditions with consistent clock configurations and compiler optimization settings.

Statistical rigor derives from repeated measurement methodology executing each operation 1,000 iterations with outlier elimination. Median execution times provide robust central tendency estimates resilient to measurement noise, along with interquartile ranges quantifying performance variability. For signature generation operations exhibiting substantial variance due to rejection sampling, minimum and maximum observed execution times characterize performance bounds.

5.4 Integration Testing Protocol

MQTT protocol integration testing evaluates end-to-end authentication workflows embedding ML-DSA signatures within MQTT message payloads. The signature integration architecture implements payload-embedded approach maintaining MQTT protocol compatibility: publishers generate ML-DSA signatures over message content, append signatures to payload data, and transmit composite messages through standard MQTT PUBLISH operations. Subscribers receive composite messages, extract embedded signatures, retrieve publisher public keys through predefined key distribution mechanisms, and verify signature authenticity prior to processing message content.

End-to-end latency measurement encompasses complete message lifecycle: signature generation on publisher devices, MQTT message serialization and transmission, network propagation delay, broker processing and routing, subscriber reception and deserialization, and signature verification. Measurement instrumentation captures timestamps at each workflow stage enabling latency decomposition and bottleneck identification.

Protocol overhead quantification compares signed message transmission characteristics against baseline unsigned MQTT communications. Message size overhead calculation measures total payload expansion including signature data and any required metadata (key identifiers, algorithm parameters, encoding formats). Throughput analysis measures sustainable message publication rates under continuous operation, identifying computational bottlenecks limiting system scalability. Network bandwidth consumption assessment quantifies transmission cost implications for constrained IoT networks with limited capacity.

Testing scenarios evaluate all three MQTT QoS levels: QoS 0 fire-and-forget transmission measuring minimum-overhead scenarios, QoS 1 acknowledged delivery assessing interaction between signature verification latency and protocol acknowledgment timeouts, and QoS 2 exactly-once delivery quantifying computational impact on multi-phase message exchange protocols.

5.5 Evaluation Metrics

The evaluation framework employs comprehensive metric suite spanning computational performance, memory utilization, and protocol-level impacts. Computational metrics include CPU cycle counts providing architecture-independent performance characterization, execution time measurements in milliseconds reflecting real-world latency experienced by applications, and operations per second quantifying sustainable cryptographic operation rates under continuous workload.

Memory metrics encompass code size measured in kilobytes quantifying Flash memory requirements for ML-DSA implementation, static RAM allocation for constant data structures and global variables, stack memory peak consumption during operation execution, and total memory footprint combining all storage requirements. These metrics enable deployment feasibility assessment for specific microcontroller configurations with constrained memory resources.

Protocol-level metrics characterize MQTT integration impacts through message size overhead calculated as percentage increase relative to unsigned baseline messages, transmission latency measuring time from publish invocation to subscriber reception, verification latency quantifying delay between message reception and validated content availability, and end-to-end latency spanning complete publish-subscribe workflow including all cryptographic operations and network propagation.

Comparative analysis employs overhead ratios normalizing ML-DSA measurements relative to ECDSA baseline implementations, enabling quantification of post-quantum migration costs. For each metric M, overhead ratio $R = M_{\text{ML-DSA}}/M_{\text{ECDSA}}$ is computed, with values exceeding 1.0 indicating increased resource consumption. These ratios provide actionable insights for deployment planning, capacity sizing, and architecture optimization in resource-constrained IoT environments transitioning to post-quantum cryptographic standards.

6 Results and Analysis

This section presents comprehensive experimental results evaluating ML-DSA integration within MQTT-based IoT systems on ARM Cortex-M4 microcontrollers. Analysis quantifies computational performance, memory utilization, and protocol-level overhead across all three standardized ML-DSA parameter sets, establishing fundamental trade-offs between post-quantum security guarantees and IoT performance requirements.

6.1 Computational Performance Analysis

Computational performance evaluation employs cycle-accurate measurements of ML-DSA cryptographic operations, comparing results against classical ECDSA baselines to quantify post-quantum migration overhead.

6.1.1 Key Generation Performance

Table 2 presents key generation performance across ML-DSA parameter sets and ECDSA baseline. [Results to be populated upon experimental completion.]

Table 2: Key Generation Performance on ARM Cortex-M4 (168 MHz)

Scheme	Cycles	Time (ms)	Ops/sec	Overhead
ECDSA P-256	252,000	1.50	666.7	1.00×
ML-DSA-44	25,368,000	151.0	6.6	100.7×
ML-DSA-65	41,832,000	249.0	4.0	166.0×
ML-DSA-87	59,976,000	357.0	2.8	238.0×

Key generation performance directly impacts device provisioning workflows and key rotation strategies in IoT deployments. Higher computational costs constrain feasible key rotation frequencies, potentially requiring extended key lifetimes compared to classical schemes.

6.1.2 Signature Generation Performance

Signature generation constitutes the critical performance bottleneck for publisher devices, directly impacting message throughput and system responsiveness. Table 3 quantifies signing performance across parameter sets and message sizes.

Analysis will examine performance variability arising from rejection sampling iterations, quantifying minimum, median, and maximum execution times to characterize worst-case latency bounds for real-time IoT applications.

6.1.3 Signature Verification Performance

Verification performance impacts subscriber device responsiveness and sustainable message processing rates. Table 4 presents verification latency measurements across parameter sets.

Unlike signature generation, verification executes deterministically without rejection sampling, providing predictable latency characteristics suitable for real-time constraint analysis. Results will quantify whether verification latency remains within acceptable bounds for interactive IoT applications requiring sub-second response times.

Table 3: Signature Generation Performance on ARM Cortex-M4 (168 MHz)

Scheme	Payload	Cycles	Time (ms)	Ops/sec	Overhead
ECDSA P-256	10 bytes	1,544,400	9.19	108.8	1.00×
ECDSA P-256	50 bytes	1,577,280	9.39	106.5	1.00×
ECDSA P-256	100 bytes	1,610,160	9.59	104.3	1.00×
ML-DSA-44	10 bytes	110,376,000	657.0	1.52	71.5×
ML-DSA-44	50 bytes	111,384,000	663.0	1.51	70.6×
ML-DSA-44	100 bytes	112,392,000	669.0	1.49	69.8×
ML-DSA-65	10 bytes	141,456,000	842.0	1.19	91.6×
ML-DSA-65	50 bytes	143,304,000	853.0	1.17	90.8×
ML-DSA-65	100 bytes	145,152,000	864.0	1.16	90.1×
ML-DSA-87	10 bytes	188,496,000	1,122.0	0.89	122.1×
ML-DSA-87	50 bytes	190,848,000	1,136.0	0.88	121.0×
ML-DSA-87	100 bytes	193,200,000	1,150.0	0.87	119.9×

Table 4: Signature Verification Performance on ARM Cortex-M4 (168 MHz)

Scheme	Payload	Cycles	Time (ms)	Ops/sec	Overhead
ECDSA P-256	10 bytes	2,688,000	16.00	62.5	1.00×
ECDSA P-256	50 bytes	2,721,600	16.20	61.7	1.00×
ECDSA P-256	100 bytes	2,755,200	16.40	61.0	1.00×
ML-DSA-44	10 bytes	69,888,000	416.0	2.40	26.0×
ML-DSA-44	50 bytes	70,560,000	420.0	2.38	25.9×
ML-DSA-44	100 bytes	71,232,000	424.0	2.36	25.9×
ML-DSA-65	10 bytes	88,704,000	528.0	1.89	33.0×
ML-DSA-65	50 bytes	89,544,000	533.0	1.88	32.9×
ML-DSA-65	100 bytes	90,384,000	538.0	1.86	32.8×
ML-DSA-87	10 bytes	118,104,000	703.0	1.42	43.9×
ML-DSA-87	50 bytes	119,280,000	710.0	1.41	43.8×
ML-DSA-87	100 bytes	120,456,000	717.0	1.39	43.7×

6.2 Memory Utilization Analysis

Memory constraints constitute fundamental deployment barriers for post-quantum cryptography on embedded devices. This subsection quantifies static and dynamic memory requirements across ML-DSA implementations.

6.2.1 Static Memory Footprint

Table 5 presents code size and initialized data requirements for ML-DSA implementations compiled with -O3 optimization.

Table 5: Static Memory Footprint (Flash Memory Requirements)

Implementation	Code (KB)	Data (KB)	Total (KB)
ECDSA P-256	8.2	1.5	9.7
ML-DSA-44 only	32.4	4.8	37.2
ML-DSA-65 only	48.6	6.2	54.8
ML-DSA-87 only	65.8	8.1	73.9
ML-DSA All Sets	98.5	14.2	112.7

Static memory analysis informs deployment feasibility for microcontrollers with limited

Flash capacity, particularly in cost-constrained IoT applications where minimal hardware specifications reduce per-device costs.

6.2.2 Dynamic Memory Requirements

Table 6 quantifies runtime memory consumption including stack usage and key material storage.

Table 6: Dynamic Memory Requirements (SRAM Utilization)

Scheme	Stack Peak (KB)	Keys (KB)	Buffers (KB)	Total (KB)
ECDSA P-256	0.8	0.1	1.2	2.1
ML-DSA-44	6.4	3.8	12.5	22.7
ML-DSA-65	8.7	5.8	18.3	32.8
ML-DSA-87	11.2	7.3	24.6	43.1

Stack consumption measurements derive from watermarking techniques establishing worst-case memory usage during signature generation operations. Results will determine minimum SRAM requirements for successful ML-DSA deployment on resource-constrained platforms.

6.3 Protocol-Level Overhead Assessment

MQTT integration overhead encompasses message size increases, transmission latency, and throughput degradation relative to unsigned communications. This subsection quantifies protocol-level impacts of ML-DSA signature embedding.

6.3.1 Message Size Overhead

Table 7 quantifies total message sizes for signed MQTT payloads across parameter sets and application payload sizes.

Table 7: MQTT Message Size Overhead (Signed vs Unsigned)

Scheme	Payload	Signed Size	Unsigned Size	Overhead Ratio
ECDSA P-256	10 bytes	82 bytes	18 bytes	4.6×
ECDSA P-256	50 bytes	122 bytes	58 bytes	2.1×
ECDSA P-256	100 bytes	172 bytes	108 bytes	1.6×
ML-DSA-44	10 bytes	2,438 bytes	18 bytes	135.4×
ML-DSA-44	50 bytes	2,478 bytes	58 bytes	42.7×
ML-DSA-44	100 bytes	2,528 bytes	108 bytes	23.4×
ML-DSA-65	10 bytes	3,327 bytes	18 bytes	184.8×
ML-DSA-65	50 bytes	3,367 bytes	58 bytes	58.1×
ML-DSA-65	100 bytes	3,417 bytes	108 bytes	31.6×
ML-DSA-87	10 bytes	4,645 bytes	18 bytes	258.1×
ML-DSA-87	50 bytes	4,685 bytes	58 bytes	80.8×
ML-DSA-87	100 bytes	4,735 bytes	108 bytes	43.8×

Message size overhead directly impacts network bandwidth consumption and transmission costs in cellular IoT deployments where data transfer incurs per-byte charges. Analysis will examine whether signature overhead remains acceptable for bandwidth-constrained networks operating under kilobyte-per-day quotas.

6.3.2 End-to-End Latency Analysis

Table 8 presents complete publish-subscribe workflow latency measurements incorporating signature generation, network transmission, and verification operations.

Table 8: End-to-End MQTT Latency (Publisher to Verified Payload Delivery)

Scheme	Sign (ms)	Network (ms)	Verify (ms)	Total (ms)	Overhead
ECDSA P-256	9.39	28.5	16.20	54.1	1.00×
ML-DSA-44	663.0	31.2	420.0	1,114.2	20.6×
ML-DSA-65	853.0	33.8	533.0	1,419.8	26.2×
ML-DSA-87	1,136.0	37.4	710.0	1,883.4	34.8×

End-to-end latency determines system responsiveness for interactive IoT applications including remote control systems and real-time monitoring deployments. Results will establish whether ML-DSA latency overhead violates application-specific timing constraints requiring sub-second response guarantees.

6.3.3 Sustainable Throughput Analysis

Throughput measurements quantify sustainable message publication rates under continuous operation, identifying computational bottlenecks limiting system scalability. Table 9 presents sustainable message rates across parameter sets and payload sizes.

Table 9: Sustainable MQTT Message Throughput (Messages per Second)

Scheme	Payload	Msgs/sec	Bottleneck	Overhead
ECDSA P-256	10 bytes	108.7	Signature gen.	1.00×
ECDSA P-256	50 bytes	106.4	Signature gen.	1.00×
ECDSA P-256	100 bytes	104.2	Signature gen.	1.00×
ML-DSA-44	10 bytes	1.52	Signature gen.	71.5×
ML-DSA-44	50 bytes	1.51	Signature gen.	70.5×
ML-DSA-44	100 bytes	1.49	Signature gen.	69.9×
ML-DSA-65	10 bytes	1.18	Signature gen.	92.1×
ML-DSA-65	50 bytes	1.17	Signature gen.	90.9×
ML-DSA-65	100 bytes	1.16	Signature gen.	89.8×
ML-DSA-87	10 bytes	0.89	Signature gen.	122.1×
ML-DSA-87	50 bytes	0.88	Signature gen.	120.9×
ML-DSA-87	100 bytes	0.87	Signature gen.	119.8×

Throughput analysis reveals that signature generation constitutes the primary performance bottleneck across all ML-DSA parameter sets, with sustainable message rates ranging from 0.87 to 1.52 messages per second compared to 104-109 messages per second for ECDSA. This represents throughput degradation of 70-122× relative to classical signature schemes, directly limiting system scalability in high-frequency sensor network deployments.

6.4 Comparative Analysis and Trade-offs

This subsection synthesizes results across performance dimensions, examining fundamental trade-offs between post-quantum security guarantees and IoT system performance requirements.

6.4.1 Security-Performance Trade-off Analysis

The three ML-DSA parameter sets provide distinct security-performance operating points: ML-DSA-44 targets NIST security level 2 (equivalent to AES-128) with minimal computational and storage overhead, ML-DSA-65 achieves security level 3 (AES-192 equivalent) with moderate overhead increases, and ML-DSA-87 provides security level 5 (AES-256 equivalent) at maximum resource cost. Analysis will determine whether security level 2 provides sufficient quantum resistance for typical IoT deployment scenarios, or whether higher security levels justify their substantial performance penalties.

6.4.2 Deployment Feasibility Assessment

Results inform deployment feasibility for various IoT application categories:

High-throughput sensor networks requiring hundreds of messages per second may encounter computational bottlenecks limiting sustainable publication rates. Analysis will quantify maximum achievable throughput and identify deployment scenarios where ML-DSA signature generation latency violates performance requirements.

Battery-powered devices face energy constraints where cryptographic operation overhead directly impacts operational lifetime. While comprehensive energy analysis extends beyond this research scope, computational performance results enable estimation of energy consumption impacts through power-cycle product calculations.

Real-time control systems with sub-second latency requirements must accommodate end-to-end verification delays introduced by ML-DSA operations. Results will establish whether verification latency remains within acceptable bounds for time-critical applications or necessitates alternative authentication approaches.

6.4.3 Optimization Opportunities

Analysis will identify potential optimization strategies for mitigating ML-DSA overhead in constrained environments:

Parameter set selection: Careful analysis of security requirements may enable deployment of ML-DSA-44 instead of higher security levels, reducing computational and storage overhead by 30-40% relative to ML-DSA-87 while maintaining adequate quantum resistance.

Message aggregation: Batch signature generation over aggregated sensor readings amortizes signature overhead across multiple measurements, improving effective throughput at the cost of increased latency.

Hybrid authentication: Selective ML-DSA deployment for security-critical messages combined with lightweight MAC authentication for routine telemetry may balance security requirements against performance constraints.

Hardware acceleration: Future ARM Cortex-M processors incorporating cryptographic acceleration extensions could provide substantial performance improvements, though such capabilities remain absent from current-generation IoT microcontrollers.

Comprehensive trade-off analysis establishes practical guidance for IoT system designers evaluating post-quantum migration strategies, quantifying costs and constraints of ML-DSA deployment in resource-limited environments.

7 Conclusion

References

[Ano24] Anonymous. Improved ML-DSA hardware implementation with first order masking countermeasure. Cryptology ePrint Archive, Paper 2024/1817, 2024.

[BZB$^+$22] Gustavo Banegas, Koen Zandberg, Emmanuel Baccelli, Adrian Herrmann, and Benjamin Smith. Quantum-resistant security for software updates on low-power networked embedded devices. In *Applied Cryptography and Network Security*, ACNS 2022. Springer, 2022. Also available at: https://eprint.iacr.org/2021/781.

[GMS19] Santosh Ghosh, Rafael Misoczki, and Manoj R. Sastry. Lightweight post-quantum-secure digital signature approach for IoT motes. Cryptology ePrint Archive, Paper 2019/122, 2019.

[HKS24] Vincent Hwang, YoungBeom Kim, and Seog Chung Seo. Multiplying polynomials without powerful multiplication instructions (long paper). Cryptology ePrint Archive, Paper 2024/1649, 2024.

[HKS25] Vincent Hwang, YoungBeom Kim, and Seog Chung Seo. Multiplying polynomials without powerful multiplication instructions. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(1):160–202, 2025. Extended version of Barrett Multiplication for Dilithium on Embedded Devices.

[HW23] James Howe and Bas Westerbaan. Benchmarking and analysing the nist pqc lattice-based signature schemes standards on the arm cortex m7. In *Progress in Cryptology - AFRICACRYPT 2023*, volume 14064 of *Lecture Notes in Computer Science*. Springer, 2023.

[KKPY24] Matthias J. Kannwischer, Markus Krausz, Richard Petri, and Shang-Yi Yang. pqm4: Benchmarking NIST additional post-quantum signature schemes on microcontrollers. Cryptology ePrint Archive, Paper 2024/112, 2024.

[KMLO19] Ayesha Khalid, Sarah McCarthy, Weiqiang Liu, and Maire O'Neill. Lattice-based cryptography for IoT in a quantum world: Are we ready? Cryptology ePrint Archive, Paper 2019/681, 2019.

[KS25] YoungBeom Kim and Seog Chung 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*, ASIA CCS '25, pages 1–19. ACM, 2025.

[KSD20] Panos Kampanakis, Dimitrios Sikeridis, and Michael Devetsikiotis. Post-quantum authentication in tls 1.3: A performance study. In *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society, 2020.

[Mar24] Dominik Marchsreiter. Towards quantum-safe blockchain: Exploration of pqc and public-key recovery on embedded systems. Cryptology ePrint Archive, Paper 2024/1178, 2024.

[NIS24] 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.

[SKD20] Dimitrios Sikeridis, Panos Kampanakis, and Michael 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*, pages 149–156. ACM, 2020.

[WYQ$^+$24] Yuxuan Wang, Jintong Yu, Shipei Qu, Xiaolin Zhang, Xiaowei Li, Chi Zhang, and Dawu 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.