Design and Implementation of a Hash-based Post-Quantum Digital Signature Scheme for Lightweight Applications

Tara Nawzad Ahmad Al Attar*, Mohammed Anwar Mohammed, Rebaz Nawzad Mohammed

Department of Computer Science, College of Science, University of Sulaimani, Sulaimaniyah, Iraq

ABSTRACT

The rapid advancement of quantum computing poses a serious threat to classical digital signature algorithms such as RSA and ECDSA, which rely on mathematical problems vulnerable to quantum attacks. Hash-based digital signatures offer a strong post-quantum alternative due to their reliance on cryptographic hash functions and their resistance to both classical and quantum adversaries. This study evaluates the feasibility of hash-based signatures in lightweight and resource-constrained environments by implementing three representative schemes: Lamport One-Time Signatures (OTS), Winternitz One-Time Signatures (WOTS), and Merkle-WOTS. The analysis focuses on key performance factors relevant to constrained devices, including key size, signature size, signing time, and verification speed. Experimental results show that while Lamport OTS provides conceptual simplicity and WOTS offers improved efficiency, the Merkle-WOTS scheme delivers the most practical balance. It supports multiple signatures under a compact public key while maintaining moderate signature sizes and competitive performance. These findings indicate that Merkle-WOTS is a strong candidate for post-quantum authentication in IoT and other lightweight embedded systems.

Index Terms: Post-Quantum Cryptography, Hash-based Signature, Winternitz One-Time Signature, Merkle Tree Authentication, Lightweight Cryptography

1. INTRODUCTION

The rapid progress of quantum computing has raised concerns about the long-term security of classical public-key cryptographic systems. Algorithms such as RSA and ECDSA, which support most digital authentication today, are expected to be broken by quantum algorithms like Shor's [1]. In response, standardization bodies such as NIST have accelerated efforts to develop post-quantum cryptographic (PQC) schemes [2].

Access this article online

DOI:10.21928/uhdjst.v10n1y2026.pp32-44	E-ISSN: 2521-4217 P-ISSN: 2521-4209

Hash-based digital signature schemes are among the strongest PQC candidates. They rely only on cryptographic hash functions, which are minimally affected by quantum attacks and benefit from decades of analysis [3].

Their simplicity and robustness make them suitable for constrained environments, although standard schemes like XMSS and LMS can still be too heavy for Internet of Things (IoT) devices and embedded systems with limited central processing unit, memory, and power [4].

This study examines the lightweight feasibility of three foundational hash-based signature schemes: Lamport One-Time Signatures (OTS), Winternitz OTS (WOTS), and the Merkle-WOTS framework. Each scheme presents different trade-offs in signature size, key management, and signing or verification cost [5].

Corresponding author's e-mail: Tara Nawzad Ahmad Al Attar, Department of Computer Science, College of Science, University of Sulaimani, Sulaimaniyah, Iraq. E-mail: tara.ahmad@univsul.edu.iq

Received: 08-11-2025 Accepted: 22-12-2025 Published: 27-01-2026

The main contributions of this work are:

- A practical Python implementation of Lamport OTS, WOTS, and Merkle-WOTS
- A unified benchmark comparing signing time, verification time, key sizes, and signature overhead
- An evaluation of their suitability for lightweight and embedded environments

This work does not propose a new algorithm. Instead, it provides a simple and lightweight implementation of the basic schemes and measures their performance using small parameters suitable for IoT and low-resource devices. Unlike the complex reference implementations of XMSS, LMS, or SPHINCS+, this study isolates the core building blocks and evaluates them in a clean, reproducible way. This offers a clear reference point for researchers and developers exploring lightweight post-quantum signatures.

Our results show that schemes such as Merkle-WOTS can provide quantum-resistant authentication with computational and memory requirements appropriate for modern constrained devices, offering practical guidance for lightweight PQC deployment.

This work contributes a lightweight and transparent implementation of Lamport OTS, WOTS, and Merkle-WOTS, evaluated using small parameter sets suitable for constrained devices. Unlike existing standards such as XMSS, LMS, and SPHINCS+, our study isolates the basic signature components and provides a clean performance benchmark that is easy to reproduce, compare, and adapt for IoT and embedded applications.

2. BACKGROUND AND RELATED WORK

2.1. Post-Quantum Motivation and Hash-based Signatures

The rise of large-scale quantum computing threatens classical digital signatures such as RSA and ECDSA, whose security depends on problems that quantum algorithms like Shor's can solve efficiently [2]. This motivates the need for digital signature schemes that remain secure in a post-quantum setting [6].

Hash-based signatures are strong candidates because they rely only on the hardness of inverting cryptographic hash functions [7], for which quantum attacks such as Grover's provide only limited speedup. Their simplicity, stateless verification, and quantum resistance make them appealing for lightweight and resource-constrained devices [8].

2.2. Lamport OTS

Lamport OTS, introduced in 1979, is one of the earliest hash-based digital signature schemes and is secured through the collision resistance of the underlying hash function [9]. Its structure is simple – signing is done by revealing selected hash preimages – but it suffers from impractically large key and signature sizes, each on the order of $O(n \cdot l)$ bits for an l-bit message and an n-bit hash output [10]. This substantial storage cost limits its practicality for lightweight or constrained devices, though Lamport OTS remains an important baseline construction in the study of hash-based signatures [11].

2.3. WOTS

WOTS improves Lamport OTS using a radix-w representation of the message digest and hash chains to compress the signature size. For an n-bit digest, the signature requires $l = l_1 + l_2$ chains, where $l_1 = \lceil n / \log_2 w \rceil$ [7]. Increasing reduces the signature length but increases the number of hash operations. Although still a one-time scheme, WOTS provides substantial size reduction and forms the basis of modern systems such as XMSS [12].

2.4. Merkle Trees and Multi-use Hash Signatures

Merkle trees enable many OTS to be grouped under a single public key. A signer generates 2^H one-time key pairs and arranges their public keys as leaves in a binary tree of height H [13]. The root becomes the global public key, and each signature includes an authentication path allowing the verifier to recompute the root. This construction greatly reduces public-key size but introduces state management to prevent reuse of one-time keys [8].

2.5. Modern Standards: XMSS and SPHINCS+

XMSS (RFC 8391) and LMS (RFC 8554) extend Merkle-tree signatures with pseudorandom key generation, improved traversal, and forward-secure properties. SPHINCS+, a NIST PQC finalist, uses hypertrees and few-time signatures to achieve stateless operation. While these schemes are mature and secure, they remain relatively heavy for severely constrained devices due to their large signatures and more complex structures [14].

2.6. Embedded and Lightweight Implementations

Several studies have evaluated hash-based signatures on embedded hardware, showing that optimized WOTS+ or XMSS implementations can achieve acceptable performance on 32-bit microcontrollers, though often requiring assembly-level tuning or hardware acceleration. Recent work also explores hybrid signatures and side-channel protections, emphasizing that efficient and secure deployment on constrained devices requires careful optimization [15].

2.7. Identified Research Gap

Existing hash-based signature standards such as XMSS, LMS, and SPHINCS+ include many added components – PRFs, large trees, and state management – that make it difficult to evaluate the raw performance of the basic signature mechanisms. There is also a lack of lightweight, open-source implementations that run easily on general-purpose systems and can be adapted to IoT devices.

This study addresses that gap by isolating Lamport OTS, WOTS, and Merkle-WOTS and evaluating them with small parameter sets suitable for constrained environments, providing clear and reproducible performance benchmarks not available in existing implementations.

3. METHODOLOGY

This section summarizes the cryptographic strategy philosophies, primitives, algorithms, and implementation optimizations involved in developing three hash-based post-quantum digital signature schemes tailored for lightweight applications. This study focuses on software-level lightweight assessment and does not include hardware measurements on embedded devices.

The implementation of Lamport OTS, WOTS, and Merkle-WOTS was carried out in Python and tested on a general-purpose PC. The goal of this setup is to provide a simple and reproducible lightweight reference for understanding the performance of these schemes. While this does not represent the behavior on embedded or low-power hardware, it offers a strong baseline for lightweight analysis. Future work will include experiments on constrained devices along with memory and energy profiling to provide a full hardware-based lightweight evaluation.

3.1. Design Goals

The future schemes are intended with the subsequent aims:

- Simplicity: Algorithms should be uncomplicated and feasible with basic hash operations, avoiding complex algebraic structures such as lattices or codes
- Memory-efficient: Given the limitations of embedded systems and IoT devices, the schemes should require minimal RAM and ROM
- Quantum Resistance: Security must be based on norms that can still grip even in the presence of large-scale quantum computers. Precisely, the dependence is on hash functions, which are only squared weakened by Grover's algorithm.

3.2. Security Model

All three schemes, Lamport OTS, WOTS, and Merkle-WOTS, originate their security from the collision resistance and prototype resistance of the fundamental hash function, SHA-256. Unlike RSA or ECC, which rely on integer factoring or discrete numerical (both broken by Shor's algorithm), these schemes are measured post-quantum protection under the statement that SHA-256 remains arithmetically unfeasible to invert or find collisions for, even with significant rivals.

The schemes aim to achieve post-quantum empirical unforgeable under chosen message attacks (EUF-CMA), lightweight efficiency in terms of time and memory, and resistance against quantum rivals through secure hash basics.

- Lamport and WOTS are EUF-CMA secure in the Random Oracle Model (ROM)
- WOTS (and by extension Merkle-WOTS) assumes one-time use per key; each message must use a fresh WOTS key.

3.3. Cryptographic Basic Uses

- SHA-256: A safe cryptographic hash function standardized by NIST, offering 256-bit output. It provides origin, accidental resistance, and assists as the cryptographic essential of all operations
- Hash Chains: Used in WOTS, where a value is frequently hashed to produce a sequence of dependent outputs. It enables a trade-off between signature size and calculation time
- Merkle Trees: A binary hash tree construction used to wrap numerous one-time public keys into a single short public key. The root serves as the global public key for Merkle-WOTS, and an authentication path is included in each signature.

3.3.1. Notation and parameters

The following notations and parameters are used consistently throughout this work:

- w: Winternitz parameter (base for encoding hash values); a higher w results in shorter signatures but increases computational cost
- l_1: Number of digits required to represent the message hash in base w
- l_2: Checksum length, added to guard against message manipulation

- l: Total number of chains used in WOTS, computed as ($l = l_1 + l_2$)
- h: Height of the Merkle tree, determining the number of one-time keys (2^H).
- n: Output length of the hash function in bits; for SHA-256, n = 256.
- SHA-256: Secure Hash Algorithm producing a 256-bit digest; used throughout all signature schemes.

3.4. Algorithm Descriptions

3.4.1. Lamport OTS

Lamport OTS is an uncomplicated hash-based signature arrangement. It operates as follows:

- Key Generation: Generate 256 pairs of 256-bit random private values. Each pair corresponds to one bit of the 256-bit hash of the message
- Public Key: Each private value is hashed using SHA-256 to produce a 256-bit public key element
- Signing: For each bit of the message hash, reveal the corresponding private value from the pair (depending on whether the bit is 0 or 1)
- Verification: Hash each part of the signature and compare it against the corresponding public key entry
- Limitation: Each key pair can be used only once, and both public and private keys are large (~16 KB each).

3.4.2. WOTS

WOTS enhances Lamport's arrangement by encoding the message hash in a higher base, reducing the number of required key elements.

- Winternitz Parameter (w): Defines the base; higher (w) decreases signature size but increases computation
- Key Generation: Generate as ($l = l_1 + l_2$) random private values. (l_1) is the number of digits in the base-w communication representation, and (l_2) accounts for a checksum that defends against communication alterations
- Signing: Each digit of the message hash regulates how many times a private value is hashed (i.e., the length of the hash chain)
- Verification: Apply the remaining hash iterations to each signature element to recreate the public key.

WOTS considerably decreases signature size and public key size compared to Lamport; nevertheless is still limited to one-time use.

3.4.3. WOTS parameter overview

WOTS considerably decreases signature size and public key size compared to Lamport, nevertheless is still limited to one-time use. The Winternitz parameter w controls the performance and security of WOTS by determining the base used for message hash encoding and the resulting number of hash chains. This work uses w = 16 to balance signature size and computational cost. The total number of chains is $l = l_1 + l_2$, where l_1 encodes the message hash and l_2 is the checksum, The WOTS signing and verification workflow is illustrated in Fig. 1.

3.4.4. Merkle-WOTS

To enable multiple signatures using WOTS, Merkle-WOTS integrates a binary Merkle tree of WOTS public keys:

- Key Generation:
  - Generate 2^h WOTS key pairs (for some tree height h)
  - Compute SHA-256 hashes of each WOTS public key to form the leaf nodes
  - Hypothesis A: Merkle tree over these leaves and compute the root, which serves as the global public key.
- Signing:
  - Use the next available WOTS key pair to sign the message
  - Include the WOTS signature and an authentication path (siblings on the Merkle tree path) in the final signature.
- Verification:
  - Verify the WOTS signature
  - Recomputed the Merkle path up to the root and compare it to the public key.

This arrangement supports up to 2^h unique signatures per master key while preserving compact global public key size (32 bytes) and adequate performance. The Merkle-WOTS tree-based signature structure is shown in Fig. 2.

3.5. Optimization Strategies

To improve performance and resource competence in limited environments, the following approaches were applied:

- Chain lengths in WOTS were effectively minimized using the Winternitz parameter (w = 16), balancing signature size and hash computations. While explicit precomputation was not implemented, the runtime behavior reflects the intended optimization
- In this study, we applied the schemes in Python and generated authentication paths for a 4-leaf Merkle tree (height h = 2). All timings and key/signature sizes reported were obtained from running the actual implementations over multiple test runs

- Memory-Efficient Base Conversions: Efficient routines are used to convert hash values into base-w formats for WOTS and to compute checksums
- Compact Hash Function: SHA-256 is used due to its wide support, hardware acceleration on some devices, and security maturity.

Authentication Path Reuse: In Merkle-WOTS, authentication paths were computed once per signature and reused during test runs to improve performance. "Persistent caching was not implemented in this version but could further reduce overhead in production environments."

4. DESIGN, IMPLEMENTATION, AND EXPERIMENTAL EVALUATION

4.1. Lamport OTS

4.1.1. Design overview

Lamport OTS is the simplest hash-based signature scheme and acts as a baseline for comparison. The private key consists of random bitstrings, and each value is hashed to form the public key. Signing reveals one value per message bit, and verification checks the corresponding hash values.

4.1.2. Key and signature characteristics

Lamport OTS provides strong security based solely on the collision resistance of the hash function. Its primary limitation is storage overhead: ≈16 KB for private/public keys and ≈8 KB for signatures, which makes it impractical for lightweight deployment.

4.1.3. Performance evaluation

Lamport achieved the fastest performance in our tests: ≈0.038 ms signing and ≈0.17 ms verification (average of five runs). The bottleneck is storage size, not computation. Although Lamport OTS provides very fast signing and verification due to its straightforward one-time structure, its key and signature sizes (≈16 KB and ≈8 KB, respectively) remain a fundamental limitation for lightweight environments. Thus, Lamport OTS serves primarily as a conceptual baseline and motivates the need for more compact constructions such as WOTS. The key sizes, signature size, and timing results for Lamport OTS are summarized in Table 1.

4.1.4. Trade-offs

Although efficient and transparent, Lamport OTS is mainly suitable as a conceptual baseline due to its large key/signature sizes. This motivates the use of more compact schemes such as WOTS.

4.2. WOTS

4.2.1. Design overview

WOTS improves Lamport by reducing key and signature sizes using base- encoding and hash chains. The Winternitz parameter controls the trade-off between computation and signature length.

4.2.2. Parameter configuration

- Hash Output: $n = 256$ bits (SHA-256)
- Winternitz Parameter: $w = 16$
- Computed Values:
  - $l_1 = \lceil n/\log_2(w) \rceil = 64$
  - $l_2 = \lceil \log_2(l_1(w\text{-}1))/\log_2(w) \rceil + 1 = 3$
  - Total *Chains*: $l = 67$

Explanation: Here, n is the hash output length, w is the Winternitz base, l_1 is the message-related chain count, and l_2 is the checksum chain count.

4.2.3. Implementation details

Each WOTS key and signature consists of 67 elements of 32 bytes. The message digest is converted to base-w, and each digit determines how many times the corresponding private key element is iteratively hashed along its chain. Verification re-applies the remaining chain steps and compares against the public key. Security relies on the collision resistance of the hash function.

4.2.4. Evaluation of WOTS

With $w = 16$, WOTS achieved ≈0.28 ms signing and ≈0.23 ms verification. Key and signature sizes were ≈2,144 bytes. These results show that WOTS provides an effective balance of storage and computation for resource-constrained platforms. However, WOTS remains one-time and must be integrated into a Merkle tree for multi-use. The performance and storage characteristics of WOTS are reported in Table 2.

4.2.5. Trade-offs

WOTS significantly reduces memory usage but remains one-time. Larger w values reduce signature size but increase chain computation. Reusing a WOTS key compromises security, requiring careful key management. WOTS must be combined with a Merkle tree for multi-signature capability.

4.3. WOTS with Merkle Tree Aggregation

4.3.1. Design overview

Merkle-WOTS extends WOTS by arranging multiple WOTS public keys as leaves of a Merkle tree. The Merkle root functions as a compact public key, and each signature includes a short authentication path.

4.3.2. Implementation characteristics
For a tree height of $h = 2$, our implementation uses four WOTS key pairs. The Merkle root is 32 bytes, and each signature contains a WOTS component plus a short authentication path.

4.3.3. Performance evaluation
Merkle-WOTS achieved ≈0.35 ms signing and ≈0.37 ms verification. The signature size was approximately 2,208 bytes, representing only a small increase over standalone WOTS. Overall, Merkle-WOTS enables multi-signature capability with minimal overhead relative to standalone WOTS. Its compact 32-byte public key, moderate signature size, and sub-millisecond signing and verification times make it a strong candidate for resource-constrained deployments. The additional cost of computing and verifying authentication paths is small and outweighed by the advantage of supporting multiple signatures per master key. Detailed performance metrics for Merkle-WOTS are presented in Table 3.

4.3.4. Performance summary of Merkle-WOTS
Merkle-WOTS achieved ≈0.35 ms signing and ≈0.37 ms verification. The signature size was approximately 2,208 bytes, representing only a small increase over standalone WOTS.

4.3.5. Trade-offs
Merkle-WOTS eliminates WOTS's one-time limitation while maintaining efficiency. Its compact public key and moderate signature size make it suitable for IoT, embedded systems, and secure firmware authentication. Among the evaluated schemes, it offers the best combination of performance, reusability, and memory efficiency.

4.4. Comparative Summary
The comparative performance analysis of the applied schemes – Lamport OTS, Winternitz OTS (WOTS), and Merkle-WOTS – establishes their discrete trade-offs in terms of size, efficiency, and scalability. The assessments summarize key parameters, including signature size, key size, signing and verification times, and reusability. All outcomes are averaged over five independent runs on Python 3.11 using the standard hashlib library on a general-purpose machine, ensuring fair and reliable measurement.

Overall, Lamport OTS suggests the modest design and fastest signing/verification times, but suffers from unfeasibly large key and signature sizes. WOTS meaningly decreases memory requirements and provides balanced efficiency, though it remains strictly one-time. Merkle-WOTS introduces modest computational overhead but enables scalable multi-use signatures under a compact public key, making it the most practical choice for lightweight applications.

4.5. Discussion
The results highlight the trade-offs between the three evaluated hash-based signature schemes under lightweight and resource-constrained conditions.

To give a broader context, it is helpful to compare these results with standard PQC schemes. XMSS (RFC 8391) and LMS (RFC 8554) usually produce 2–4 KB signatures and rely on larger Merkle trees with PRFs and state handling, while SPHINCS+ generates even larger signatures (8–30 KB) due to its stateless design. In contrast, the lightweight schemes tested in this work use much smaller parameters and simpler structures, resulting in faster operations and smaller keys. Although the systems differ in design and are not directly comparable, this context positions our lightweight measurements relative to established PQC methods.

Lamport OTS is the simplest scheme but suffers from large key and signature sizes (≈16 KB keys and ≈8 KB signatures), limiting its suitability for constrained devices. However, it provides a clear baseline for understanding hash-based signatures. It was also the fastest scheme in our tests (≈0.038 ms signing, ≈0.17 ms verification), showing that storage – not speed – is the main bottleneck.

WOTS improves storage efficiency through base-w encoding and hash chains. With w = 16, it achieved a balanced performance (≈0.28 ms signing, ≈0.23 ms verification) and significantly smaller key and signature sizes. Its main limitation is its one-time nature, which requires integration with Merkle trees for multi-message signing.

Merkle-WOTS enables reusable signing capabilities while keeping overhead low. With a height-2 tree, the scheme achieved ≈0.35 ms signing and ≈0.37 ms verification, a compact 32-byte public key, and a moderate signature size (~2,208 bytes). This makes it the most practical option among

TABLE 1: Performance of Lamport OTS implementation

Metric	Value	Notes
Public key size	16 KB	512 hashes×32 bytes
Private key size	16 KB	512 random values×32 bytes
Signature size	8 KB	256 values×32 bytes
Signing time	≈0.038 ms	Average over 5 runs
Verification time	≈0.17 ms	Average over 5 runs
Scalability	1 message	Strict one-time use

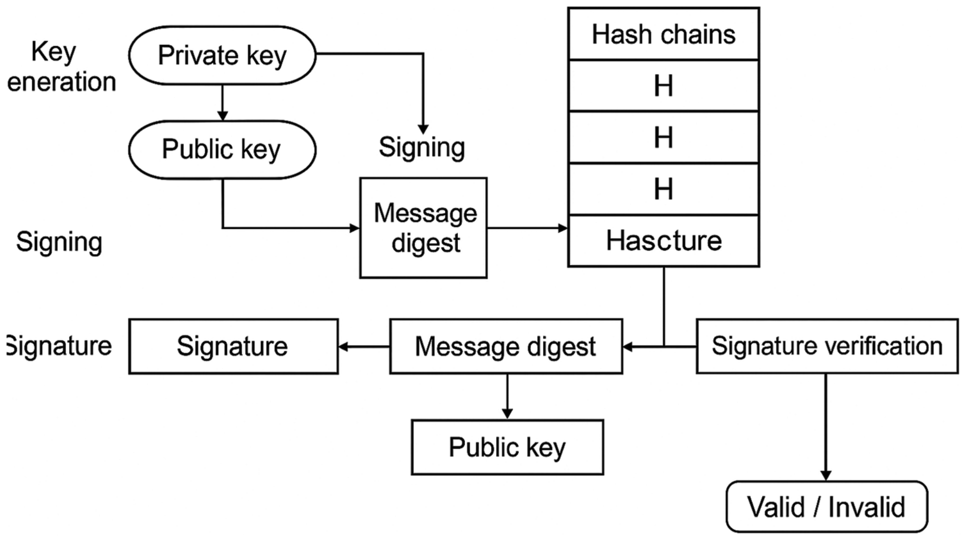

Fig. 1. Winternitz One-Time Signature (WOTS) Workflow.

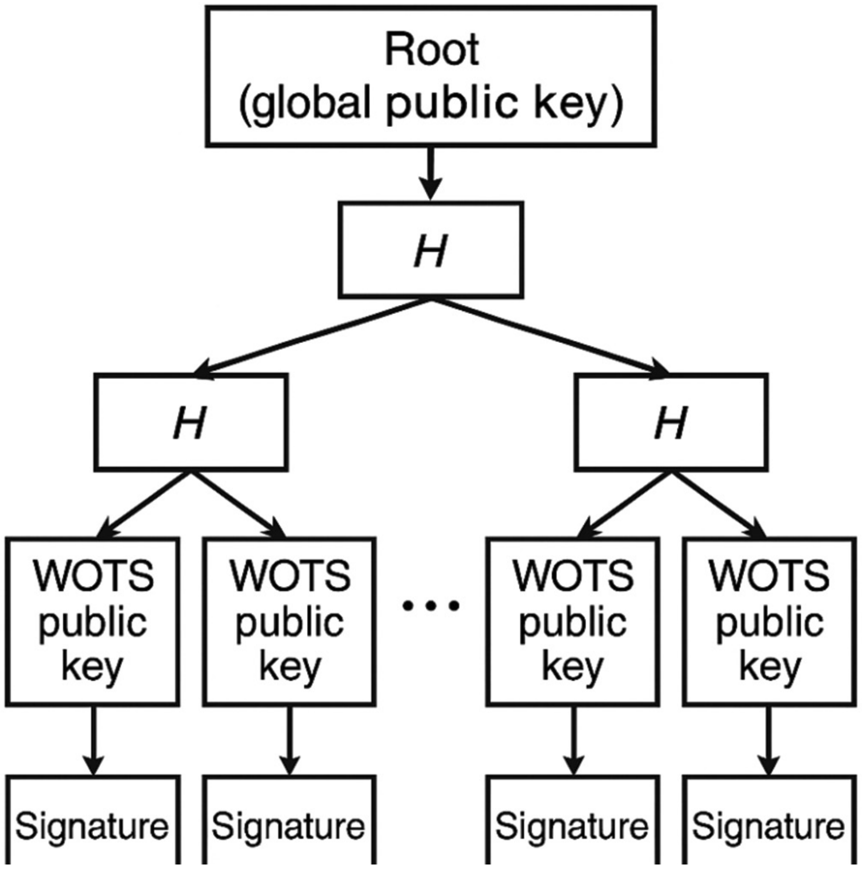

Fig. 2. Merkle-WOTS tree-based signature structure.

the three for IoT and embedded systems, where multi-use and memory efficiency are important.

All schemes were tested over five runs, and the averaged results were used to ensure consistency. Overall, Lamport

TABLE 2: Performance of WOTS implementation

Scheme	Signing time (ms)	Verification time (ms)	Signature size (bytes)	Public key size (bytes)	Private key size (bytes)
Lamport OTS	0.038	0.17	8192	16384	16384
WOTS (w=16)	0.28	0.23	2144	2144	2144
WOTS+Merkle (h=2)	0.35/2.46*	0.37/2.25*	2208	32	2144

TABLE 3: Performance of Merkle-WOTS implementation

Metric	Value	Notes
Public key size	32 bytes	Merkle tree root
Private key size	2,144 bytes	Same as WOTS per leaf
Signature size	2,208 bytes	= WOTS signature 2,144 B+auth path h·32 B with h=2
Signing time	≈ 0.35 ms	Average over 5 runs
Verification time	≈ 0.37 ms	Average over 5 runs
Scalability	2^h	h=2 → 4 signatures supported

and base WOTS provide insight into fundamental trade-offs, while Merkle-WOTS offers the most suitable balance of performance, reusability, and resource efficiency for lightweight post-quantum applications.

Table 4 situates our lightweight schemes relative to established post-quantum signature standards. XMSS and LMS produce signatures in the 2–4 KB range with millisecond-level signing and verification times, but require stateful key management. SPHINCS+, while stateless, has significantly larger signatures (8–30 KB) and incurs higher computation costs.

In contrast, our Merkle-WOTS implementation achieves comparable or smaller signature sizes (~2.2 KB), a compact 32-byte public key, and sub-millisecond timings, highlighting its suitability for constrained environments.

Fig. 3 illustrates the relative computational cost of the three evaluated schemes. Lamport OTS achieves the lowest signing and verification times due to its simple one-time structure, while Merkle-WOTS introduces a small overhead from authentication path computations. Despite this cost, all schemes operate in the sub-millisecond range, confirming lightweight suitability.

As shown in Fig. 4, Lamport OTS has significantly larger key and signature sizes compared to WOTS and Merkle-WOTS. This reinforces Lamport's role as a conceptual baseline rather than a practical lightweight solution. Merkle-WOTS offers the smallest public key (32 bytes) with only a marginal increase in signature size over WOTS.

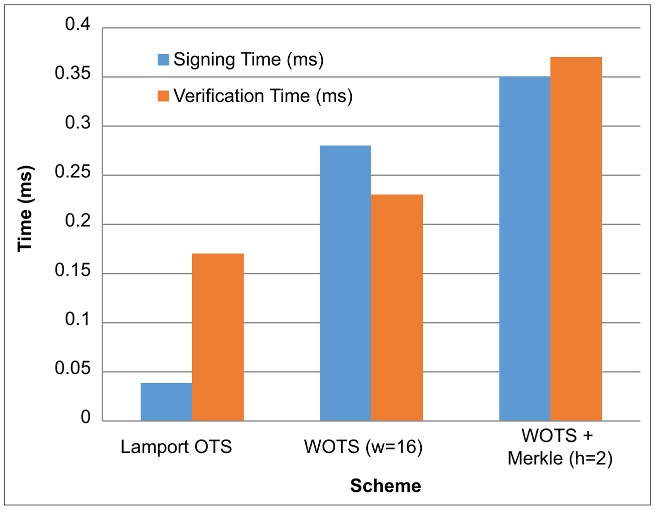

Fig. 3. Signing and verification time comparison.

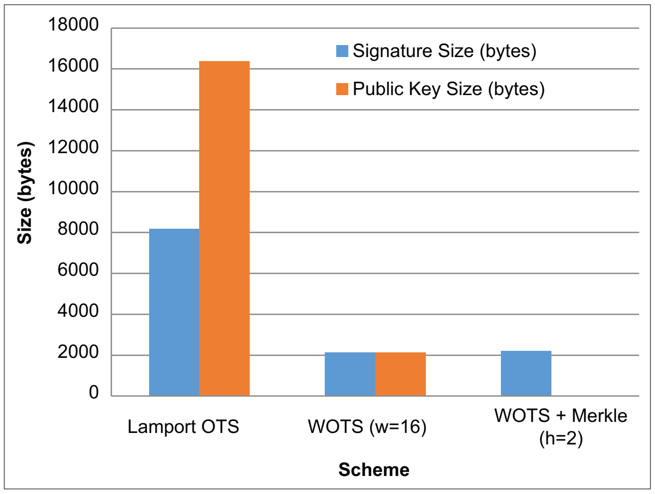

Fig. 4. Performance and size comparison of hash-based digital signature and public key size.

Here are two comparative charts that visually summarize the experimental results from your implementation, as illustrated in Table 5:

1. Signing and Verification Time (in milliseconds)
2. Signature and Public Key Size (in bytes)

TABLE 4: Comparison of lightweight schemes with standard PQC hash-based signatures

Scheme	Signature size	Public key size	Signing time	Verification time	Notes
XMSS (RFC 8391)	2–3 KB	1 KB	2–10 ms	2–10 ms	Stateful, uses WOTS+and Merkle tree
LMS (RFC 8554)	2–4 KB	32 B	1–8 ms	1–8 ms	Stateful; NIST-approved
SPHINCS+(NIST PQC)	8–30 KB	32 B	10–50 ms	10–50 ms	Stateless, secure but heavy
Lamport OTS	8 KB	16 KB	0.038 ms	0.17 ms	One-time; large keys
WOTS (w=16)	2.1 KB	2.1 KB	0.28 ms	0.23 ms	One-time; lightweight
Merkle-WOTS (h=2)	2.2 KB	32 B	0.35 ms	0.37 ms	Multi-use; most practical lightweight option

TABLE 5: Hash signature performance charts

Scheme	Signing time (ms)	Verification time (ms)	Signature size (bytes)	Public key size (bytes)
Lamport OTS	0.038	0.17	8192	16384
WOTS (w=16)	0.28	0.23	2144	2144
WOTS+Merkle (h=2)	0.35	0.37	2208	32

4.6. Security Discussion

To support the post-quantum motivation of the evaluated schemes, it is important to provide a summary of their security strength. The security of Lamport OTS, WOTS, and Merkle-WOTS relies entirely on the hardness of preimage, second-preimage, and collision resistance of the underlying hash function. When using SHA-256, classical security corresponds to 256-bit preimage resistance and 128-bit collision resistance. Under quantum adversaries using Grover's algorithm, the effective strength is reduced to approximately 128-bit preimage security, which remains sufficient for many lightweight PQC applications [16].

The Winternitz parameter w also influences the security margin: smaller w results in longer signatures but tighter security bounds, while larger w increases efficiency but slightly reduces security. Merkle-WOTS inherits the security of WOTS and additionally achieves a many-time security structure based on the Merkle tree, where the tree height w determines the total number of secure signatures [17].

Although lightweight schemes such as Lamport and WOTS lack the extended security components found in XMSS or SPHINCS+, they still provide strong quantum-resistant guarantees suitable for constrained platforms, provided each key is used only once, and the underlying hash function remains secure [18].

5. EXPERIMENTAL EVALUATION AND RESULTS

All performance tests were carried out on a controlled setup using a general-purpose computer. The hardware and software specifications of the test platform are summarized in Table 6.

TABLE 6: Hardware and software configuration used for performance benchmarking

Component	Specification
Processor	Intel Core i5-1135G7 @ 2.40 GHz
Memory	8 GB RAM
OS	Ubuntu 22.04 LTS (64-bit)
Python Version	Python 3.11
Libraries	hashlib, os, time
Timer	time.perf_counter()

The three schemes – Lamport OTS, WOTS, and WOTS with Merkle trees – were implemented in Python using only standard libraries to maintain portability and repeatability. Although the platform is not an embedded device, it provides a stable baseline for comparing its performance. Key generation, signing, and verification were measured using Python's high-resolution time. perf_counter(), with each operation executed 5 times and averaged to reduce system noise. This approach ensures consistent results and meaningful comparisons across the schemes.

5.1. Repeatability and Transparency

To support repeatability, all source code and test scripts are available in the GitHub repository, allowing other researchers to reproduce the results, adapt the setup to their own hardware, or extend the work to additional hash-based schemes.

5.2. Future Hardware Validation

While this study provides a baseline on general-purpose hardware, future work will evaluate the schemes on embedded platforms such as ARM Cortex-M, RISC-V, and ESP32. Testing on these devices will allow a more accurate assessment of memory usage, energy consumption, and real-time suitability for lightweight IoT and edge-computing environments.

TABLE 7: Performance summary of signature schemes

Scheme	Signing time	Verification time	Signature size	Public key size	Private key size
Lamport OTS	0.038 ms	0.17 ms	8192 B	16384 B	16384 B
WOTS (w=16)	0.28 ms	0.23 ms	2144 B	2144 B	2144 B
WOTS+Merkle (h=2)	0.35 ms	0.37 ms	2208 B	32 B	2144 B (per leaf)

In the WOTS+Merkle scheme, N represents the number of WOTS leaf keys. In this implementation, *n*=4, corresponding to a Merkle tree of height 2

TABLE 8: High-level comparison of digital signature schemes in terms of size, time, and deployment suitability

Scheme	Signature size	Private key size	Public key size	Signing time	Verification time	Reusability	Suitability (summary)
Lamport OTS	8192 B	16384 B	16384 B	0.038 ms	0.17 ms	One-time	Fastest; impractical key sizes
WOTS (w=16)	2144 B	2144 B	2144 B	0.28 ms	0.23 ms	One-time	Compact and efficient
WOTS+Merkle (h=2)	2208 B	2144 B (per leaf)	32 B	0.35 ms	0.37 ms	2h per root (multi-use)	Scalable; tiny public key, practical

5.3. Benchmark Results and Analysis

To evaluate the practical viability of the implemented signature schemes, we measured key performance indicators related to computation and storage, including signing time, verification time, and the sizes of signatures and keys. All values represent the average of five runs on the benchmark platform described in this section, and the comparative results are summarized in Table 7.

5.4. Comparative Analysis

Memory and storage efficiency – mainly signature and key sizes – is a critical factor for cryptographic deployment in constrained environments such as implanted devices, sensor nodes, and mobile platforms. The evaluated schemes show significant differences in space usage, as high-level comparison of all evaluated schemes in terms of size, performance, and deployment suitability is shown in Table 8:

- Lamport OTS produces a signature of 8,192 bytes and a public key of 16,384 bytes, creating substantial storage and communication overhead. These large sizes make it unsuitable for devices with limited memory or narrow bandwidth
- WOTS (w = 16) greatly improves efficiency, reducing both key and signature sizes to about 2,144 bytes each. This represents roughly a 74% reduction and makes WOTS a more practical choice when compact signatures are required
- WOTS + Merkle Tree further enhances space efficiency with a constant 32-byte public key derived from the Merkle root. While the authentication path increases the signature to 2,208 bytes, this small overhead is acceptable given the added scalability and ability to support multiple signatures.

Figs. 5-7 compare signature, private-key, and public-key sizes, respectively. The results emphasize the efficiency of WOTS and Merkle-WOTS, with Merkle-WOTS achieving the smallest public key due to its Merkle-root structure. Lamport OTS consistently appears as an outlier because of its inherently large key material.

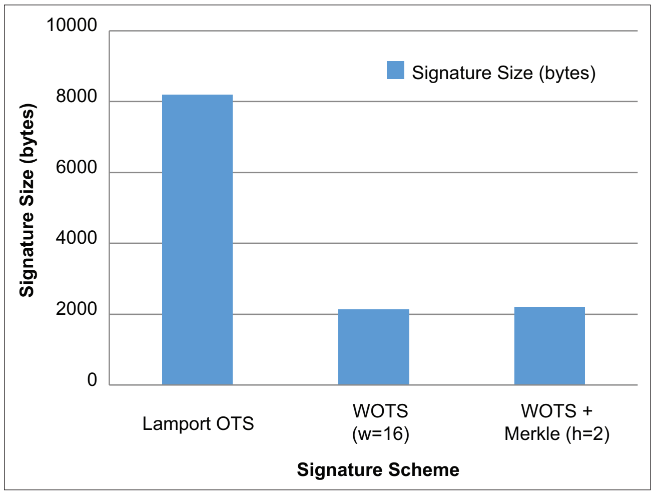

Fig. 5. Comparison of signature sizes across schemes.

Figs. 8 and 9 confirm that all schemes produce fast operations suitable for constrained devices. The timing differences between WOTS and Merkle-WOTS are minimal, showing that Merkle-tree authentication imposes limited overhead.

5.5. Scalability and Suitability for Lightweight Devices

Beyond space efficiency, implementation speed is also important for assessing practicality in constrained settings. All three schemes – Lamport OTS, WOTS, and WOTS with Merkle Trees – achieve signing and verification in the millisecond range, making them suitable for devices with moderate processing power, including microcontrollers, smart sensors, and mobile platforms.

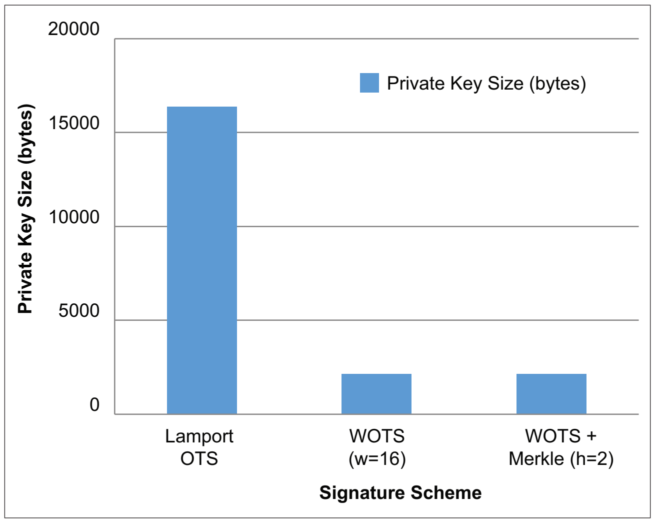

Fig. 6. Comparison of private key size across schemes.

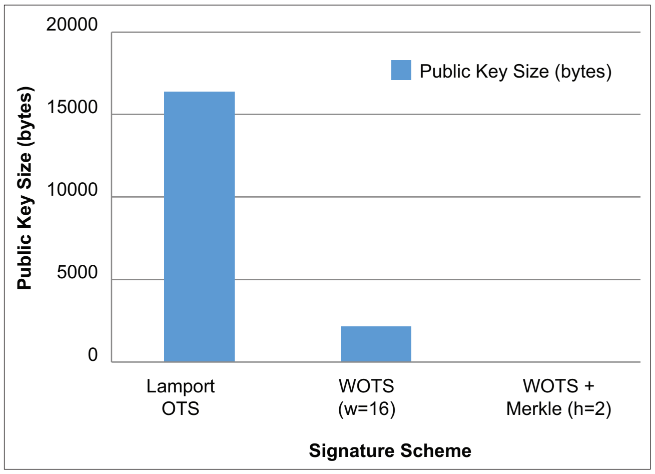

Fig. 7. Comparison of public key size across schemes.

5.6. Performance Summary

All three schemes demonstrate sub-millisecond to low-millisecond performance, confirming their suitability for real-time systems and resource-aware cryptographic protocols. Theoretically, both Lamport OTS and WOTS are EUF-CMA secure in the ROM, offering strong guarantees based on the hardness of inverting the hash function. However, WOTS requires unique message input per key, so reusing a key compromises security and reinforces its one-time nature.

Together with the experimental results, these observations show that while Lamport and WOTS mainly serve as theoretical baselines, Merkle-WOTS extends these foundations into a scalable and practical option for lightweight post-quantum applications.

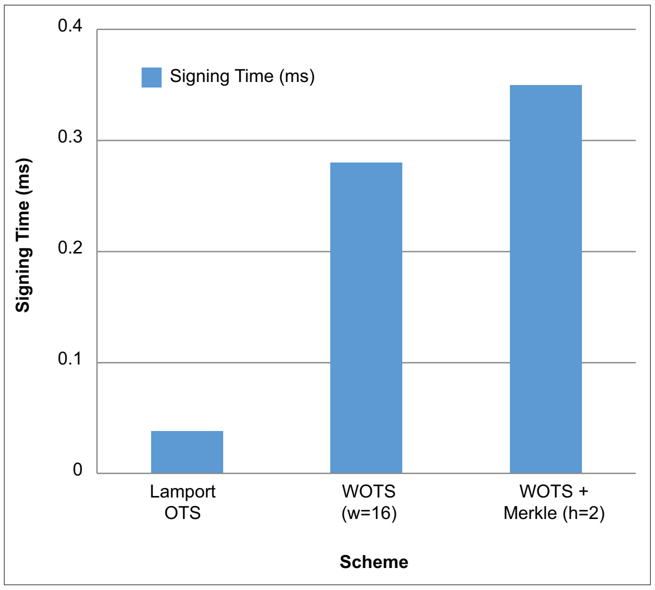

Fig. 8. Comparison of signing times across schemes.

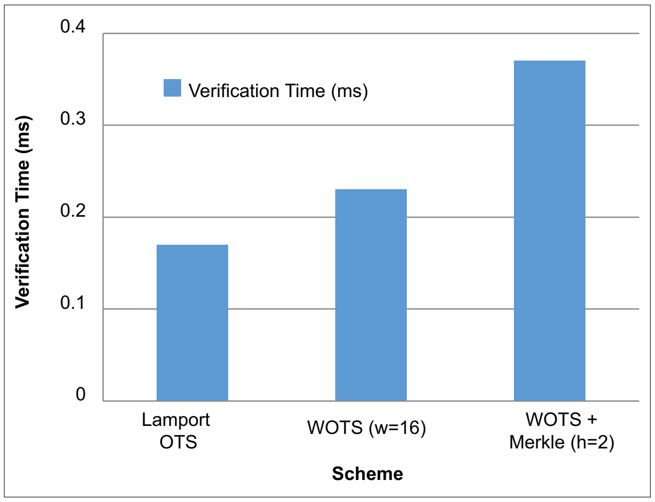

Fig. 9. Comparison of verification times across schemes.

5.7. Practical Implications

The experimental results show that WOTS + Merkle provides a practical and deployable hash-based signature scheme for resource-constrained environments such as embedded systems, IoT devices, and smart platforms. It achieves a strong balance between post-quantum security, signature compactness, and multi-message scalability, making it suitable for lightweight applications that require both efficiency and long-term reliability.

Key application scenarios include:

- Secure boot and firmware integrity checks on microcontrollers

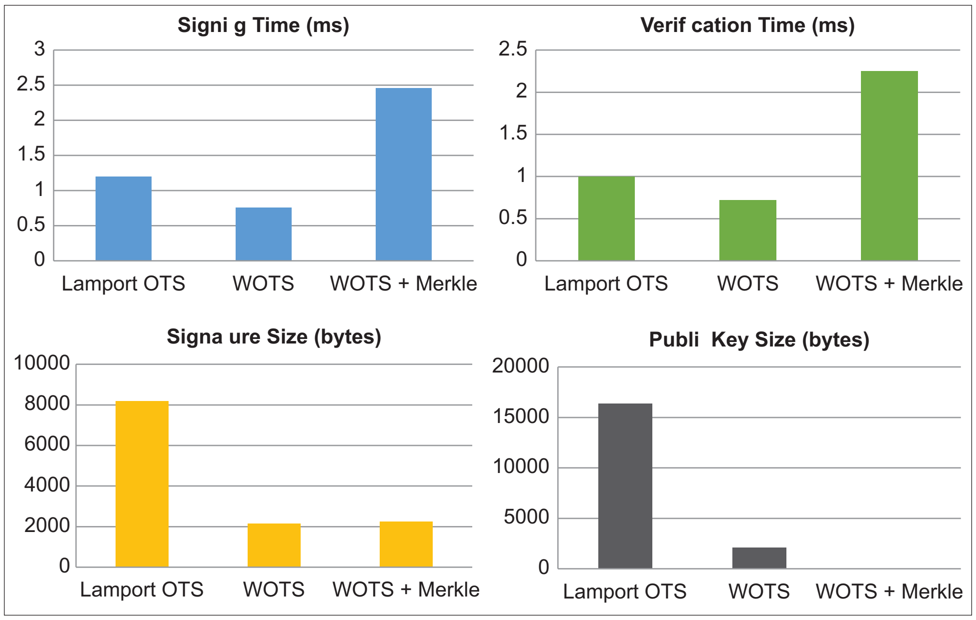

Fig. 10. Performance comparison chart.

- Authenticated OTA updates in industrial and automotive IoT
- Remote device verification over untrusted networks
- Broadcast authentication in sensor or mesh networks

Compared to other post-quantum options, Merkle-WOTS offers:
- A very small public key (32 bytes), ideal for limited storage
- A moderate signature size (~2.2 KB), suitable for wireless transmission
- Fast signing (~0.35 ms) and verification (~0.37 ms) on general-purpose hardware
- Scalability to 2^h signatures without regenerating keys.

Although Merkle-WOTS is slower than base WOTS due to authentication path generation and verification, this added cost is modest. The ability to use multiple signatures under a compact public key outweighs the small latency increase, making Merkle-WOTS the most practical candidate among the evaluated schemes for lightweight post-quantum deployment.

The reliance on standard hash primitives such as SHA-256 ensures easy software integration and wide platform compatibility, even on systems without cryptographic accelerators or advanced math libraries. These findings strongly support the practical viability of deploying hash-based digital signatures in future post-quantum embedded and lightweight security systems.

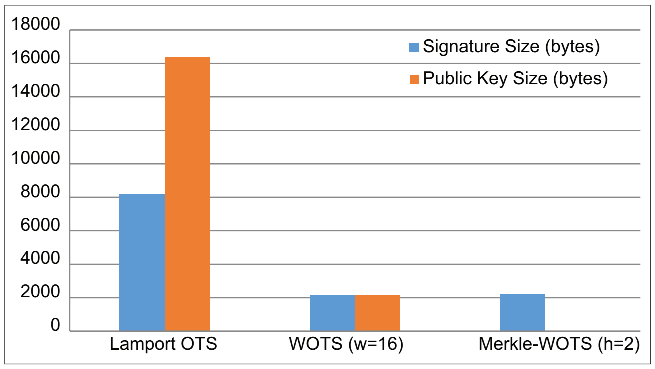

Fig. 11. Deployment-oriented comparison of signature and public key sizes.

Figs. 10 and 11 summarize the deployability of the schemes by juxtaposing signature size, key size, and performance. Merkle-WOTS offers the best balance and is therefore the most practical candidate for post-quantum authentication in embedded systems.

6. DISCUSSION

The results demonstrate clear differences between the three evaluated hash-based signature schemes in terms of size, performance, and scalability. Lamport OTS is simple and secure but impractical for constrained devices due to its large key and signature sizes, despite being the fastest scheme tested. WOTS reduces storage requirements and maintains good performance through base-w encoding, but its strict one-time nature limits standalone use. Merkle-WOTS offers the most practical balance: it supports multiple signatures under a compact public key and keeps signature sizes moderate, with only a small overhead from authentication path processing.

Overall, Merkle-WOTS provides the best suitability for lightweight and embedded environments, showing that hash-based signatures can meet post-quantum authentication needs using only standard hash functions.

7. CONCLUSION AND FUTURE WORK

This work implemented and evaluated three fundamental hash-based signature schemes – Lamport OTS, WOTS, and Merkle-WOTS – with a focus on lightweight and resource-constrained environments. The results confirm the expected trade-offs between security, performance, and storage. Lamport OTS offers conceptual simplicity but remains impractical due to its large key and signature sizes. WOTS provides significant reductions in storage and maintains fast signing and verification, but remains limited by its strict one-time nature. Merkle-WOTS achieves the most practical balance, supporting multiple signatures under a compact public key while maintaining moderate signature size and competitive performance.

Overall, the findings show that Merkle-WOTS is a strong candidate for post-quantum authentication in IoT and embedded systems, offering scalability and efficiency using only standard hash primitives.

8. FUTURE WORK

Future directions include implementing the schemes in lower-level languages (e.g., C or Rust) and benchmarking them on real embedded hardware such as ARM Cortex-M or RISC-V microcontrollers. Additional work involves measuring memory and energy usage under constrained conditions, integrating Merkle-WOTS into lightweight protocols such as MQTT, CoAP, or DTLS, and exploring advanced security features such as forward-security, key evolution, and hybrid verification models for frequent-use scenarios.

9. REFERENCES

[1] C. K. Gitonga. "The impact of quantum computing on cryptographic systems: Urgency of quantum-resistant algorithms and practical applications in cryptography". *European Journal of Information Technologies and Computer Science*, vol. 5, no. 1, pp. 1-10, 2025.

[2] G. Nkulenu. "Quantum computing: The impending revolution in cryptographic security". *International Journal of Multidisciplinary Research and Growth Evaluation*, vol. 5, pp. 1137-1149, 2024.

[3] S. Suhail, R. Hussain, A. Khan and C. S. Hong. "On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions". *IEEE Internet of Things Journal*, vol. 8, no. 1, pp. 1-17, 2020.

[4] P. Kampanakis and S. Fluhrer, "*LMS vs XMSS: Comparion of two hash-based signature standards*," Cryptology ePrint Archive, vol. Paper 2017/349, 2017. Available from: https://eprint.iacr.org/2017/3490

[5] T. N. A. Al Attar, M. A. Mohammed and R. N. Mohammed. "Exploring post-quantum cryptography: Evaluating algorithm resilience against global quantum threats". *UHD Journal of Science and Technology*, vol. 9, no. 1, pp. 18-28, 2025.

[6] T. N. A. Al Attar and R. N. Mohammed. "Optimization of lattice-based cryptographic key generation using genetic algorithms for post-quantum security". *UHD Journal of Science and Technology*, vol. 9, no. 1, pp. 93-105, 2025.

[7] L. Li, X. Lu and K. Wang. "Hash-based signature revisited". *Cybersecurity*, vol. 5, no. 1, p. 13, 2022.

[8] E. Fathalla and M. Azab. "Beyond classical cryptography: A systematic review of post-quantum hash-based signature schemes, security, and optimizations". *IEEE Access*, vol. 12, pp. 175969-175987, 2024.

[9] H. I. Kaplan, *"General Review of Hash-Based Signatures,"* Cryptology ePrint Archive, vol. 2025/1398, 2025. Available from: https://ia.cr/2025/1398

[10] V. Srivastava, A. Baksi and S. K. Debnath, *"An overview of hash based signatures,"* Cryptology ePrint Archive, vol. 2023/411, no. 2023, 2023. Available from: https://ia.cr/2023/411

[11] P. Mazza. "*Temporal Resource Comparison between Classical Asymmetric Cryptosystems and Post-Quantum Alternatives*". Politecnico di Torino, Italy, 2025.

[12] K. Zhang, H. Cui and Y. Yu. "*Revisiting the Constant-Sum Winternitz One-Time Signature with Applications to and XMSS*". In: Annual International Cryptology Conference, Springer, pp. 455-483, 2023.

[13] T. G. Tan, P. Szalachowski and J. Zhou. "Challenges of post-quantum digital signing in real-world applications: A survey". *International Journal of Information Security*, vol. 21, no. 4, pp. 937-952, 2022.

[14] P. A. Mohan. "*Hash-based Digital Signatures-A Tutorial Review*". In: 2023 IEEE International Conference on Public Key Infrastructure and its Applications (PKIA), IEEE, pp. 1-8, 2023.

[15] P. Tandel and J. Nasriwala. "Secure authentication framework for IoT applications using a hash-based post-quantum signature scheme". *Service Oriented Computing and Applications*, pp. 1-12, 2024, 2024.

[16] A. Joshi, P. Bhalgat, P. Chavan, T. Chaudhari and S. Patil. "*Guarding Against Quantum Threats: A Survey of Post-Quantum Cryptography Standardization, Techniques, and Current Implementations*". In: International Conference on Applications and Techniques in Information Security, Springer, pp. 33-46, 2024.

[17] C. Majenz, C. M. Manfouo and M. Ozols. "Quantum-Access Security of the Winternitz One-Time Signature Scheme". [arXiv Preprint]; 2021.

[18] C. M. Pacurar, R. Bocu and M. Iavich. "An analysis of existing hash-based post-quantum signature schemes". *Symmetry*, vol. 17, no. 6, p. 919, 2025.