Post-quantum Verkle signature scheme with SHAKE-based deterministic randomness and quantum entropy

Maksim Iavich[1,*,†]

[1]Department of Computer Science, Caucasus University, Paata Saakadze Str., 1, Tbilisi, 0102, Georgia

Abstract

This paper presents a novel Verkle signature scheme that integrates a SHAKE-based deterministic random bit generator (DRBG) and quantum random number generator (QRNG) to achieve efficient, scalable, and post-quantum secure digital signatures. The proposed design addresses the inefficiencies of traditional one-time signature schemes, which suffer from high storage and key management overheads. By using a QRNG for seed initialization and SHAKE-based PRNG for deterministic key generation, the system produces on-demand one-time key pairs without requiring persistent storage. These keys are structured within a Verkle tree, leveraging vector commitments to reduce proof sizes and optimize verification. The resulting architecture offers strong security against quantum attacks, forward secrecy, and reduced computational and memory costs, making it suitable for deployment in constrained or large-scale environments. Our approach ensures compatibility with post-quantum cryptographic requirements while maintaining practical efficiency.

Keywords

post-quantum cryptography, security, memory optimization, cryptographic applications, Merkle tree hash, Verkle tree, vector commitment schemes, lattices,

1. Introduction

The rise of quantum computing introduces a fundamental shift in the landscape of information security. Many widely deployed cryptographic protocols rely on the presumed difficulty of problems such as integer factorization and discrete logarithms. These assumptions underpin schemes like RSA, DSA, and elliptic curve cryptography (ECC), which are at the heart of modern secure communication systems. However, advances in quantum algorithms, most notably Shor's algorithm, have demonstrated that these mathematical problems can be efficiently solved using quantum computers. As a result, current public key infrastructures would become insecure in the presence of a scalable quantum adversary [1, 2, 3].

This imminent threat has initiated a global effort to design post-quantum cryptographic (PQC) algorithms that can remain secure even against adversaries equipped with quantum capabilities [4, 5, 6]. A wide range of candidate systems has emerged, including schemes based on lattices, error-correcting codes, multivariate polynomials, and hash-based constructions. Despite their cryptographic strength, many of these proposals pose challenges in terms of implementation, including increased computational overhead, large key sizes, or high memory demands. These limitations are especially critical in constrained environments such as embedded systems, edge devices, and sensor networks. Hash-based signature schemes offer a compelling alternative for post-quantum security. Their security relies solely on the strength of cryptographic hash functions, which are expected to remain resistant to both classical and quantum attacks. Merkle tree-based constructions, in particular, have a long-standing reputation for robustness. However, traditional Merkle trees often suffer from large proof sizes, significant storage requirements for key pairs, and inefficient update mechanisms, making them less practical for modern applications that demand compact and scalable authentication [7, 8, 9]. To address these issues, this work introduces a new lightweight digital signature scheme that utilizes Verkle trees. Verkle trees

CH&CMiGIN'25: Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks, June 20–22, 2025, Kyiv, Ukraine

*Corresponding author.

†These authors contributed equally.

miavich@cu.edu.ge (M. Iavich)

0000-0002-3109-7971 (M. Iavich)

represent an evolution of Merkle trees, replacing hash-based commitments with vector commitments. This substitution allows for much smaller proof sizes and more efficient membership verification, particularly when managing large datasets. By reducing both the communication and verification complexity, Verkle trees are better suited for high-throughput and low-resource cryptographic systems [10, 11, 12].

In addition to structural improvements through Verkle trees, we propose a novel approach to randomness and key generation. Instead of relying solely on stored key material, our scheme introduces a hybrid mechanism that combines a true quantum random number generator (QRNG) with a SHAKE-based deterministic random bit generator (DRBG) [13, 14, 15]. The QRNG serves as a source of high-entropy seed material derived from quantum physical processes, ensuring unpredictability. This entropy is then used to initialize the SHAKE-based DRBG, which generates pseudorandom values for on-demand key creation. The SHAKE function, part of the SHA-3 family approved by NIST, supports arbitrary output lengths and provides strong resistance to quantum attacks.

This approach resolves the limitations of key storage in one-time signature schemes by allowing reproducible key generation from a secure seed. Keys are generated only when needed, enabling the system to maintain strong cryptographic guarantees while minimizing storage and memory overhead. Moreover, using SHAKE-based DRBG as a core component aligns the construction with established post-quantum standards, increasing the scheme's compatibility with evolving cryptographic frameworks.

The primary objectives of this research are outlined as follows:

- To propose a digital signature scheme that integrates Verkle trees for reduced proof size and faster verification compared to traditional Merkle constructions.
- To design a key generation and randomness framework that combines quantum entropy from QRNG with deterministic pseudorandomness from SHAKE-based DRBG, enhancing unpredictability and scalability.
- To eliminate the need for persistent storage of large key sets in one-time signature systems by enabling dynamic and deterministic key generation.
- To demonstrate how the use of vector commitments within Verkle trees supports efficient aggregation and verification, reducing the computational burden on verifying parties.
- To lay out a cryptographically secure framework suitable for long-term post-quantum applications, including digital identity, secure messaging, and blockchain infrastructure, without the inclusion of implementation-specific performance measurements.

This paper presents a forward-looking signature construction designed to withstand the impact of quantum computing while remaining efficient and scalable. By combining Verkle trees with hybrid randomness generation, the scheme achieves both theoretical soundness and practical adaptability. It is well-positioned for adoption in scenarios where security, compactness, and quantum resistance are essential.

2. Hash-based one-time signatures and Merkle tree schemes

2.1. One-time signature schemes

One-time signature (OTS) schemes provide secure digital signatures for a single message per key pair. A classical example is the Lamport-Diffie scheme, which uses hash functions and random bit strings for secure message authentication. The private key consists of $2n$ random bit strings:

$$Y = (y_0[0],\, y_0[1],\, \ldots,\, y_{n-1}[0],\, y_{n-1}[1])\,, \quad y_i[j] = f(x_i[j]). \tag{1}$$

To sign a message M, a digest $din0, 1^n$ is computed using a hash function g:

$$X = (x_0[0],\, x_0[1],\, x_1[0],\, x_1[1],\, \ldots,\, x_{n-1}[0],\, x_{n-1}[1]) \in \{0,1\}^{n\times 2}. \tag{2}$$

The public key is derived by applying a one-way function f to each element of the private key:

$$d = g(M) = (d_0,\, d_1,\, \ldots,\, d_{n-1})\,. \tag{3}$$

The signature consists of one value from each key pair, selected according to the corresponding bit in the message digest:

$$\text{sign} = (x_0[d_0],\, x_1[d_1],\, \ldots,\, x_{n-1}[d_{n-1}])\,. \tag{4}$$

Verification involves applying the function f to each value in the signature and comparing the result with the corresponding public key entries:

$$\big(f(\text{sign}_0),\, f(\text{sign}_1),\, \ldots,\, f(\text{sign}_{n-1})\big) = (y_0[d_0],\, y_1[d_1],\, \ldots,\, y_{n-1}[d_{n-1}])\,. \tag{5}$$

Although secure, this approach is limited by the fact that each key can only be used once. To improve efficiency, schemes like Winternitz OTS (W-OTS) allow multiple bits to be signed with fewer hash operations.

2.2. Merkle tree signature schemes

To overcome the key reuse limitation of OTS schemes, Merkle trees are used to scale signature systems [16, 17, 18]. In a Merkle signature scheme (MSS), a tree of height H is built from the public keys of 2^H one-time signatures:

$$(X_j, Y_j), \quad j = 0, 1, \ldots, 2^H - 1. \tag{6}$$

Each internal node is computed by hashing its child nodes:

$$v_i = h\,(v_{2i},\, v_{2i+1})\,. \tag{7}$$

The root of the tree becomes the overall public key of the system. When a message is signed, a one-time key is used along with an authentication path-a sequence of sibling nodes that allows the verifier to reconstruct the root of the tree.

To verify the signature, the verifier checks the one-time signature, reconstructs the root using the authentication path, and compares it with the known public key.

While Merkle trees offer scalability, they come with trade-offs. The signer must store a large number of private keys, and the signature includes multiple hash values in the authentication path, which increases its size as the tree height grows.

Figure 1 shows an example of a Merkle tree with height of $H = 3$.

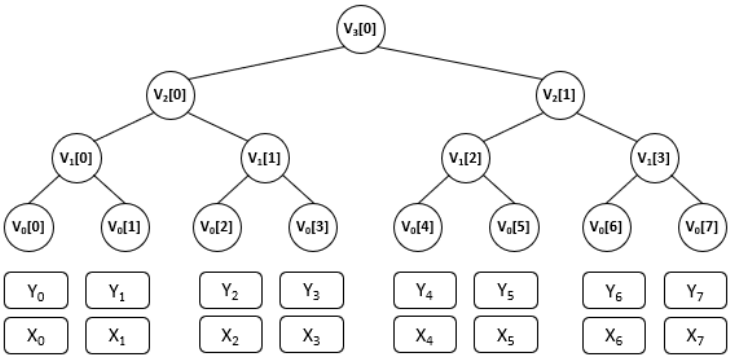

Figure 1: Merkle tree.

3. Verkle trees for efficient commitment and proof structures

Cryptographic protocols designed for the post-quantum era must balance strong security with practical efficiency, particularly in terms of memory usage and verification speed. Verkle trees provide an effective

alternative to Merkle trees by enabling much smaller proofs while maintaining integrity and verifiability. This makes them especially useful in systems where compactness and verification speed are critical, such as blockchain networks and resource-constrained cryptographic devices.

3.1. Concept and structure

A Verkle tree is a hierarchical data structure that uses vector commitments to commit to a large set of values in a compact and efficient way. Each internal node in the tree aggregates several child values or commitments into a single root using a commitment scheme that allows selective opening. Assume a node holds k values denoted by:

$$f_0,\ f_1,\ \ldots,\ f_{k-1}. \tag{8}$$

A commitment is created over this collection using a vector commitment function:

$$VC = \text{VectorCommit}\left(f_0,\ f_1,\ \ldots,\ f_{k-1}\right). \tag{9}$$

This commitment serves as the representation of all values at that node. To prove that a particular value f_i is part of the committed set, an opening proof is generated that reveals only f_i and proves its inclusion without disclosing the other values.

In the same manner, internal nodes can commit to the vector commitments of their children. This recursive structure continues up the tree, and the final commitment at the top level becomes the root:

$$\text{Root} = \text{VectorCommit}\left(VC_1,\ VC_2,\ \ldots,\ VC_m\right). \tag{10}$$

This root serves as the digest of the entire Verkle tree and is used for verification in applications such as digital signatures or data integrity proofs.

3.2. Comparison to Merkle trees

Verkle trees offer significant advantages compared to classical Merkle trees. In a Merkle tree, to prove that a leaf is part of the tree, the prover must provide all sibling hashes along the path to the root. This leads to a proof size that grows with the height of the tree. In contrast, Verkle trees allow each internal node to cover many values. As a result, the depth of the tree decreases, and proofs consist of fewer elements. The proof includes one opening per vector commitment along the path, rather than full sibling sets.

Let d be the branching factor of the tree and h its height. In this case, proof size scales approximately with $\log_d n$, where n is the number of committed values. A larger branching factor d reduces the depth of the tree and, consequently, the number of openings required in a proof.

This approach brings practical benefits:

- Significantly smaller proofs;
- Faster verification time due to fewer operations;
- Better scalability in large systems.

3.3. Relevance to post-quantum systems

Verkle trees are well aligned with the goals of post-quantum cryptography. Vector commitments can be constructed using assumptions that are considered secure against quantum adversaries, such as lattice-based problems or hash-based constructions.

By incorporating vector commitments into tree structures, Verkle trees enable signature schemes and data authentication mechanisms that are both space-efficient and resistant to quantum attacks. This makes them a strong candidate for future cryptographic frameworks, especially in environments where memory, computation, and bandwidth are limited.

4. Lattice-based vector commitments

Vector commitments (VCs) allow committing to a sequence of values in such a way that individual elements can later be revealed and verified without disclosing the rest. This makes VCs particularly useful for privacy-preserving systems, cryptographic accumulators, authenticated databases, and other public verification schemes. One important challenge in this field is designing VCs that remain secure in the post-quantum era.

Recent research has explored how VCs can be constructed from lattice-based assumptions, especially the Short Integer Solution (SIS) problem. These lattice-based schemes support stateless updates, meaning proofs and commitments can be updated efficiently without requiring the full vector. Unlike traditional Merkle trees, where updating a single value may require recomputing many hashes, SIS-based constructions offer better scalability.

However, early designs suffered from inefficiencies in large dimensions, as the cost for both the committer and verifier scaled poorly [19, 20]. To overcome this, a tree-like construction has been proposed that allows the commitment dimension to be extended hierarchically while preserving compact proofs. By selecting a suitable branching factor and tree height, the system can maintain short proof sizes, enabling practical use even in constrained environments.

The improved structure relies on specialized algebraic tools such as a "gadget" matrix and trapdoor sampling techniques. These components allow efficient preimage sampling and compact proof generation. Although the setup phase requires some trusted parameters and private initialization, the result is a VC system with small, verifiable proofs and support for stateless updates, all while being grounded in post-quantum security.

Overall, lattice-based VCs offer a promising path toward scalable and quantum-resistant commitment schemes, making them valuable for advanced cryptographic protocols like Verkle tree-based digital signatures.

5. Dynamic key generation using PRNG and QRNG in Verkle trees

In signature schemes like the Merkle Signature Scheme (MSS), each one-time signature requires a unique key pair. For a tree of height H, this results in 2^H private keys, which becomes impractical to store in many real-world systems. To reduce this burden, a deterministic pseudo-random number generator (PRNG) can be used to generate keys on demand from a single initial seed. This approach eliminates the need to store all private keys. The PRNG takes an input seed $S_{in} \in \{0,1\}^n$ and returns both a new seed and a pseudorandom output:

$$\text{PRNG} : S_{\text{in}} \mapsto (\text{RAND},\ S_{\text{out}}). \tag{11}$$

Starting from a randomly chosen seed S_0, a sequence of one-time signature seeds is derived iteratively:

$$(S_{\text{OTS}_j},\ S_{j+1}) = \text{PRNG}(S_j), \quad \text{for } j = 0, 1, \ldots, 2^H - 1. \tag{12}$$

Each seed $\mathrm{S_{OTS_j}}$ is then used to deterministically generate the components of the j-th one-time signature key:

$$(x_0,\ x_1,\ \ldots,\ x_{t-1}) = \text{PRNG}\left(S_{\text{OTS}_j}\right). \tag{13}$$

This method ensures that only the current seed is needed to compute the next key, significantly reducing memory requirements. Keys can be reconstructed exactly when needed-during signing or verification, without the need to store them persistently.

The same technique can be applied in Verkle trees, which also associate keys with leaf nodes. Each key-value pair can be generated dynamically using a seeded PRNG. This enables efficient signature generation and verification without the overhead of managing a large static key store.

To enhance security and randomness, the PRNG can be seeded using output from a Quantum Random Number Generator (QRNG). QRNGs use physical quantum phenomena to produce truly random bits,

making them ideal for cryptographic applications. Seeding the PRNG with high-entropy quantum input ensures strong unpredictability in key generation, reinforcing the system's resistance against quantum attacks. In practice, the PRNG is used in two phases: first, during key generation to initialize the tree structure, and second, at signature time to regenerate the specific one-time key needed. This method allows the system to remain lightweight and scalable while maintaining strong post-quantum security properties. As Verkle trees evolve, replacing classical commitment schemes with lattice- or hash-based post-quantum alternatives becomes critical to future-proofing such systems against quantum threats.

6. SHAKE-based deterministic random bit generator (DRBG)

To ensure strong post-quantum security in our digital signature scheme, we adopt a SHAKE-based Deterministic Random Bit Generator (DRBG) as the core pseudorandom number generator. SHAKE functions, part of the SHA-3 family standardized by NIST (FIPS 202), offer extendable output lengths and excellent resistance against both classical and quantum cryptanalytic attacks. This makes SHAKE-based DRBGs a natural choice for cryptographic applications in the post-quantum era. Unlike traditional DRBGs based on block ciphers, SHAKE-based DRBGs operate entirely through sponge construction. Given an initial seed with sufficient entropy, the SHAKE function absorbs the seed into its internal state and then generates arbitrary-length pseudorandom output. This output is deterministic and indistinguishable from true randomness, provided the seed is unpredictable.

The advantages of SHAKE-based DRBGs include simplicity of design, strong resistance to differential and linear cryptanalysis, and a reduced need for reseeding compared to block cipher-based generators. Their extendable-output nature allows for efficient key material expansion, nonce generation, and on-demand derivation of one-time keys without introducing unnecessary overhead.

In our architecture, the SHAKE-DRBG is seeded once with high-entropy input from a quantum random number generator (QRNG). From that point, the SHAKE function provides pseudorandom outputs used for generating key material and signature components. This hybrid setup ensures both unpredictability (due to QRNG) and determinism with performance efficiency (due to SHAKE). The use of SHAKE-based DRBG not only eliminates reliance on AES or counter-based mechanisms but also simplifies implementation in memory-constrained environments, making it especially suitable for post-quantum digital signature schemes deployed in IoT, embedded devices, or blockchain networks.

7. Generating a one-time key pair using a quantum-seeded SHAKE-based DRBG

To support digital signature schemes in a post-quantum environment, we rely on a secure and efficient pseudorandom number generator (PRNG). For this purpose, we utilize a SHAKE-based DRBG (Deterministic Random Bit Generator), seeded with high-entropy input from a Quantum Random Number Generator (QRNG). This hybrid approach combines the cryptographic strength of the SHA-3 sponge construction with the unpredictability offered by quantum phenomena. SHAKE-based DRBGs operate by absorbing a seed into the internal state of the sponge and then squeezing out arbitrary-length output. Unlike counter-based constructions that rely on block ciphers, SHAKE-based generators derive security from their cryptographic sponge design, eliminating the need for key or counter management. The seed is generated by a QRNG, ensuring high entropy and true randomness, which is critical for secure key generation in the quantum era.

Let the SHAKE-based DRBG be defined as follows: given an initial n-bit seed $\mathrm{QRNG}_{\mathrm{in}}$ from a QRNG, the generator produces a random output RAND of n bits and an updated internal seed $\mathrm{QRNG}_{\mathrm{out}}$ of n bits.

$$\text{SHAKE-DRBG} : \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^n, \quad \mathrm{QRNG}_{\mathrm{in}} \mapsto (\mathrm{RAND},\, \mathrm{QRNG}_{\mathrm{out}}). \tag{14}$$

The generation of one-time keys in the scheme begins with a uniformly random seed QRNG_0. We then compute a sequence of derived seeds $\mathrm{QRNG} - \mathrm{OTS}_{\mathrm{j}}$ for each j in the range $0 \leq j < 2^H$, where

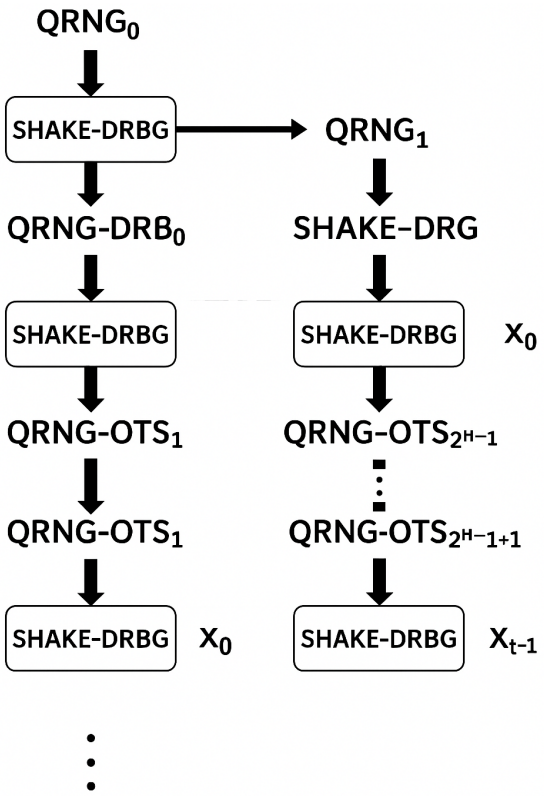

Figure 2: Generating a one-time signing key using SHAKE-based DRBG

H defines the tree height. These seeds are recursively derived by applying the $SHAKE - DRBG$ function:

$$(\text{QRNG-OTS}_j,\ \text{QRNG}_{j+1}) = \text{SHAKE-DRBG}(S_j), \quad 0 \leq j < 2^H. \tag{15}$$

Each seed $\text{QRNG} - \text{OTS}_\text{j}$ then generates the corresponding one-time signature key X_j, composed of t blocks of n-bit strings:

$$(x_i,\ \text{QRNG-OTS}_j) = \text{SHAKE-DRBG}(\text{QRNG-OTS}_j), \quad i = t{-}1, \ldots, 0. \tag{16}$$

Each invocation of CTR_DRBG results in an update of the seed $\text{QRNG} - \text{OTS}_\text{j}$, demonstrating that the signature key X_j can be derived simply from QRNG_j. Furthermore, the computation of $\text{QRNG} - \text{OTS}_\text{j}$ defines a new seed $\text{QRNG}_{\text{j}+1}$ for the signature key $X(j+1)$. The image below illustrates this process of generating a one-time signature key using CTR_DRBG. This process starts with $QRNG_0$, the secret key, which has a length of n. The seeds $\text{QRNG}_{\text{j}+1}$ determined during the creation of the signature key X_j subsequently replace $QRNG_0$.

This structure ensures that the entire one-time key X_j is deterministically generated from a single seed. Furthermore, since each use of $SHAKE - DRBG$ updates the seed, it eliminates the risk of seed reuse. The updated seed $\text{QRNG}_{\text{j}+1}$ becomes the basis for generating the next key $\text{X}_{\text{j}+1}$, ensuring that each signature key remains isolated and securely derived.

The process is visualized in Figure 2, where $QRNG_0$ initiates the secure generation chain. At each step, a new seed is derived and used to generate the next one-time signing key. Because SHAKE allows flexible-length output, both the key and its expansion are efficiently supported without additional cryptographic primitives.

By seeding with quantum-generated entropy and leveraging the cryptographic properties of SHAKE, this key generation process achieves a high level of security and efficiency. The scheme guarantees that past keys cannot be reconstructed even if the current state is exposed, supporting forward security. Moreover, because SHAKE-DRBG is stateless and deterministic, no persistent state must be retained between key generations, reducing attack surfaces and simplifying implementation.

This method ensures that all one-time signatures generated before termination remain verifiable and secure. Since each signing key is used only once and is derived from a seed that cannot regenerate previous keys, the scheme maintains strong forward secrecy and post-quantum resistance.

8. The improved Verkle signature scheme using SHAKE-based PRNG and QRNG

To achieve both post-quantum security and implementation efficiency, we propose an improved Verkle signature scheme that combines a quantum random number generator (QRNG) with a SHAKE-based deterministic random bit generator (DRBG). This hybrid approach supports secure, on-demand key generation, minimizes persistent storage requirements, and is well-suited for real-world cryptographic systems operating under memory or processing constraints.

8.1. System overview

The design begins with a high-entropy seed generated using a QRNG. Quantum-generated randomness ensures unpredictability beyond the capabilities of classical pseudorandom sources. This seed is then used to initialize a SHAKE-based PRNG, selected for its post-quantum security assumptions, flexible output length, and efficient sponge function structure. The PRNG is employed in two critical stages:

1. Key Generation Phase – The SHAKE-based DRBG produces one-time signature key pairs deterministically from the initial seed.
2. Signing Phase – A fresh one-time signature key is derived for each message, ensuring signature uniqueness and forward secrecy.

By avoiding the need to store all keys and instead regenerating them from the seed, the system improves both scalability and security.

8.2. Key generation

Let $QRNG_0$ be the initial n-bit seed produced by the quantum random number generator. This seed initializes the SHAKE-based PRNG, which then outputs a sequence of one-time key pairs (X_j, Y_j), where:

- X_j is the private signing key,
- Y_j is the corresponding public verification key,
- $0 \leq j < 2^H$, with $H \in \mathbb{N},\ H \geq 2$ determining the tree height.

Each new key pair is generated on-demand and can be recomputed by replaying the PRNG from the original seed up to index j, eliminating the need for persistent storage.

8.3. Signing process

To sign a message M, the following steps are executed:

1. Compute the message digest $d = g(M)$.
2. Use the next available one-time key X_s (from the PRNG) to sign the digest.
3. Include the corresponding public key Y_s and the authentication path from Y_s to the Verkle tree root.
4. The final signature includes: sign $= (X_s,\ Y_s,\ \text{path},\ d)$.

This ensures that each signature is unique and cryptographically bound to a specific one-time key.

8.4. Verification process

Verification involves:

1. Recomputing the message digest $d = g(M)$.
2. Verifying the one-time signature using Y_s.
3. Authenticating Y_s using the provided path in the Verkle tree.
4. Checking that the root commitment matches the known public key.

Only if all conditions are met is the signature accepted as valid.

8.5. Verkle tree integration

Each one-time public key Y_j becomes a leaf in the Verkle tree. Internal nodes aggregate commitments using a vector commitment scheme. The root commitment of the tree acts as the global public key.

During signing:

- The authentication path from leaf Y_s to the root is provided.
- This allows the verifier to confirm Y_s's inclusion in the tree and link the signature to the public key.

This structure enables the system to store just one root public key while supporting a large number of one-time signatures.

9. Conclusions

This paper introduces an enhanced Verkle signature scheme that addresses the critical needs of modern cryptographic systems in a post-quantum era. The architecture integrates a SHAKE-based Deterministic Random Bit Generator (DRBG) with entropy sourced from a Quantum Random Number Generator (QRNG), providing a robust foundation for generating unpredictable and cryptographically secure keys. Our approach eliminates the dependency on large-scale key storage, a major drawback in traditional one-time signature schemes such as Merkle Signature Scheme (MSS), by allowing the deterministic reproduction of one-time keys on demand.

By employing Verkle trees, which offer compact proofs and verification efficiency via vector commitments, the scheme drastically reduces the size of signatures and associated metadata. This is particularly beneficial in bandwidth-constrained or resource-limited environments, where large signature structures pose both computational and storage challenges.

The scheme also ensures:

- Forward security: every key pair is used only once, preventing reuse and enabling strong resistance to key compromise.
- Scalability: through stateless, on-the-fly key generation and tree-based verification, the architecture scales gracefully with the number of signatures and users.
- Quantum resistance: SHAKE-based DRBGs belong to the SHA-3 family, which are known to resist attacks by quantum adversaries. Coupled with true quantum entropy via QRNG, this hybrid design strengthens both the unpredictability and long-term viability of the system.
- Efficient verification and update paths: the use of vector commitments within Verkle trees supports batch verification and reduced overhead when validating multiple signatures, without sacrificing integrity or security.

Acknowledgments

This work was supported by the Shota Rustaveli National Science Foundation of Georgia (SRNSFG) N FR-24-15007.

Declaration on Generative AI

The author has not employed any Generative AI tools.

References

[1] M. AbuGhanem, Ibm quantum computers: Evolution, performance, and future directions, The Journal of Supercomputing 81 (2025) 687.

[2] I. Abdikhakimov, The interplay of quantum computing, blockchain systems, and privacy laws: Challenges and opportunities, Elita.uz – Elektron Ilmiy Jurnal 2 (2024) 1–12.

[3] M. Iavich, The evolution of digital signatures: From classical to post-quantum, 2024. Unpublished manuscript or preprint.

[4] D. Joseph, et al., Transitioning organizations to post-quantum cryptography, Nature 605 (2022) 237–243.

[5] M. Iavich, The evolution of digital signatures: From classical to post-quantum, 2024. Duplicate of iavich2024a; consider merging or citing only once.

[6] Z. Hu, S. Gnatyuk, T. Okhrimenko, S. Tynymbayev, M. Iavich, High-speed and secure prng for cryptographic applications, International Journal of Computer Network and Information Security 12 (2020) 1–10. doi:10.5815/ijcnis.2020.03.01.

[7] V. Srivastava, A. Baksi, S. K. Debnath, An overview of hash based signatures, Cryptology ePrint Archive (2023).

[8] L. Li, X. Lu, K. Wang, Hash-based signature revisited, Cybersecurity 5 (2022) 13.

[9] T. Lange, Hash-based signatures, in: Encyclopedia of Cryptography, Security and Privacy, Springer Nature Switzerland, Cham, 2025, pp. 1110–1112.

[10] J. Kuszmaul, Verkle trees, Verkle Trees 1 (2019).

[11] M. Iavich, T. Kuchukhidze, R. Bocu, A post-quantum digital signature using verkle trees and lattices, Symmetry 15 (2023) 2165.

[12] Y. Averyanova, et al., UAS cyber security hazards analysis and approach to qualitative assessment, in: S. Shukla, A. Unal, J. V. Kureethara, D. K. Mishra, D. S. Han (Eds.), Data Science and Security, volume 290 of *Lecture Notes in Networks and Systems*, Springer, Singapore, 2021, pp. 258–265. doi:10.1007/978-981-16-4486-3_28.

[13] M. Schöffel, J. Feldmann, N. Wehn, Efficient hardware implementation of constant time sampling for hqc, 2023. arXiv:arXiv:2309.16493.

[14] M. Siswanto, B. Rudiyanto, Designing of quantum random number generator (qrng) for security application, in: 2017 3rd International Conference on Science in Information Technology (ICSITech), IEEE, 2017.

[15] O. Solomentsev, M. Zaliskyi, O. Kozhokhina, T. Herasymenko, Efficiency of data processing for uav operation system, in: IEEE 4th International Conference on Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), 2017, pp. 27–31. doi:10.1109/APUAVD.2017.8308769.

[16] G. Becker, Merkle signature schemes, merkle trees and their cryptanalysis, Technical Report Technical Report 12, Ruhr-University Bochum, 2008.

[17] M. Szydlo, Merkle tree traversal in log space and time, in: EUROCRYPT 2004, Springer, 2004.

[18] M. Iavich, et al., Improved post-quantum merkle algorithm based on threads, in: Advances in Computer Science for Engineering and Education III, Springer, 2021, pp. 403–410.

[19] O. C. Okoro, M. Zaliskyi, S. Dmytriiev, O. Solomentsev, O. Sribna, Optimization of maintenance task interval of aircraft systems, International Journal of Computer Network and Information Security 14 (2022) 77–89. doi:10.5815/ijcnis.2022.02.07.

[20] N. S. Kuzmenko, I. V. Ostroumov, K. Marais, An accuracy and availability estimation of aircraft positioning by navigational aids, in: IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC), 2018, pp. 36–40. doi:10.1109/MSNMC.2018.8576276.