applied sciences

MDPI

Article

Post-Quantum Delegated Proof of Luck for Blockchain Consensus Algorithm

Hyunjun Kim, Wonwoong Kim, Yeajun Kang, Hyunji Kim and Hwajeong Seo *

Division of IT Convergence Engineering, Hansung University, Seoul 02876, Republic of Korea
* Correspondence: hwajeong@hansung.ac.kr; Tel.: +82-2-760-8033

Abstract: The advancements in quantum computing and the potential for polynomial-time solutions to traditional public key cryptography (i.e., Rivest–Shamir–Adleman (RSA) and elliptic-curve cryptography (ECC)) using Shor's algorithm pose a serious threat to the security of pre-quantum blockchain technologies. This paper proposes an efficient quantum-safe blockchain that incorporates new quantum-safe consensus algorithms. We integrate post-quantum signature schemes into the blockchain's transaction signing and verification processes to enhance resistance against quantum attacks. Specifically, we employ the Falcon signature scheme, which was selected during the NIST post-quantum cryptography (PQC) standardization process. Although the integration of the post-quantum signature scheme results in a reduction in the blockchain's transactions per second (TPSs), we introduce efficient approaches to mitigate this performance degradation. Our proposed post-quantum delegated proof of luck (PQ-DPoL) combines a proof of luck (PoL) mechanism with a delegated approach, ensuring quantum resistance, energy efficiency, and fairness in block generation. Experimental results demonstrate that while post-quantum cryptographic algorithms like Falcon introduce larger signature sizes and slower processing times, the PQ-DPoL algorithm effectively balances security and performance, providing a viable solution for secure blockchain operations in a post-quantum era.

Keywords: post-quantum blockchain; consensus algorithm; quantum computing; post-quantum cryptography

check for updates

Citation: Kim, H.; Kim, W.; Kang, Y.; Kim, H.; Seo, H. Post-Quantum Delegated Proof of Luck for Blockchain Consensus Algorithm. *Appl. Sci.* **2024**, *14*, 8394. https://doi.org/10.3390/app14188394

Academic Editor: Christos Bouras

Received: 22 July 2024
Revised: 13 September 2024
Accepted: 16 September 2024
Published: 18 September 2024

1. Introduction

Blockchain is a technology that securely records and manages transactions on a decentralized network, bringing innovation to various industries such as finance, supply chain management, healthcare, real estate, and energy [1,2]. A key element of blockchain security is the elliptic-curve cryptography (ECC)-based signature method, which is widely used for transaction signing and verification due to its efficiency and relatively small key size. ECC has become the standard for blockchain implementations because it provides strong security with less computational overhead compared to other cryptographic methods.

However, recent advancements in quantum computing present a significant threat to the security of ECC-based systems. Quantum computers leverage principles of quantum mechanics to perform calculations at speeds unattainable by classical computers. Shor's algorithm [3] can solve discrete logarithm problems on elliptic curves in polynomial time, effectively breaking ECC's security and rendering current blockchain systems vulnerable to quantum attacks that could compromise transaction integrity.

As quantum computers approach practical use, transitioning to post-quantum cryptography (PQC) becomes crucial to ensure the long-term security of cryptographic systems. PQC algorithms are designed to withstand quantum attacks. However, integrating PQC into blockchain systems poses challenges [4]. These issues are particularly problematic for real-time blockchain applications, where the performance degradation severely limits transaction processing speeds. Post-quantum signature schemes result in larger signatures and slower signing and verification times, which not only reduce transaction throughput and computational efficiency but also increase data transmission costs and cause longer propagation delays across the network. As a result, the system requires more storage and bandwidth. This delay in propagating transactions across the network slows down the validation and verification processes, which in turn reduces the overall throughput of the blockchain.

To address these concerns, this paper proposes post-quantum delegated proof of luck (PQ-DPoL): a novel and efficient blockchain consensus algorithm. PQ-DPoL is designed to be secure and efficient even in the era of quantum computing, with the aim of fair participation among nodes. Previous consensus algorithms, such as proof of work (PoW), are energy-intensive and prone to centralization due to rising mining costs, while proof of stake (PoS) and delegated proof of stake (DPoS) also carry risks of centralization and collusion among delegated nodes, potentially undermining blockchain decentralization.

To address these issues, PQ-DPoL combines the randomness and fairness inherent in proof of luck (PoL) with a delegated approach, enhancing both efficiency and security. Unlike PoL, which relies on trusted execution environments (TEEs) to ensure randomness and security, PQ-DPoL eliminates this dependency. Instead, it leverages verifiable random function (VRFs) to guarantee fairness and randomness in the selection process without relying on potentially vulnerable TEEs. Furthermore, PQ-DPoL employs the Falcon signature scheme, which was chosen during the NIST post-quantum cryptography standardization process, to enhance resistance against quantum attacks. While post-quantum cryptographic algorithms like Falcon introduce performance overheads, PQ-DPoL effectively balances security and performance, providing a viable solution for secure blockchain operations in a post-quantum era.

This paper is structured as follows: Section 2, Related Work, reviews current blockchain consensus algorithms and the advancements in post-quantum cryptography (PQC), highlighting the need for quantum-resistant solutions. Section 3, PQ-DPoL Algorithm Description, explains the design and implementation of the post-quantum delegated proof of luck (PQ-DPoL) algorithm. Section 4, Experimental Results, presents the performance evaluation of PQ-DPoL, comparing transaction processing speed (TPSs), signing and verification speeds, and energy consumption with traditional ECC-based systems. Section 5, Conclusions and Future Work, summarizes the key findings, discusses the contributions of the PQ-DPoL algorithm, and suggests potential future research directions for further optimization and exploration of additional PQC schemes.

Extension from WISA'23 (Kim et al.)

This current work is indeed a substantially enlarged version of our previous paper presented at WISA 2023. For reference, the WISA paper can be found at [5].

The current version of the paper presents a more efficient approach than the one proposed in [5]. While the previous paper focused on integrating the Dilithium signature method into the blockchain consensus algorithm, this paper switches to the Falcon signature method, which offers the advantage of smaller signature sizes and faster verification speeds. Additionally, our previous paper concentrated on the integration of the Dilithium approach within a TEE-based proof of luck (PoL) framework. In contrast, this paper employs verifiable random functions (VRFs) to enhance the randomness and fairness of the delegate node selection process, thereby mitigating the risks associated with centralization and hardware dependencies present in the original PoL mechanism.

2. Related Work

2.1. Blockchain Consensus Algorithms

Consensus algorithms are essential for ensuring that blockchain networks operate securely and without a central authority. They allow nodes to agree on the validity of transactions and the addition of new blocks, maintaining the integrity and consistency of the network. Without consensus algorithms, the blockchain would be vulnerable to attacks, inconsistencies, and double-spending, undermining its reliability and trustworthiness.

Proof of work (PoW) involves participants solving complex mathematical problems, with the first to find a solution creating a block and earning rewards. PoW prevents double-spending and network attacks by requiring participants to solve complex mathematical problems to validate transactions. However, this process is highly energy-intensive and requires substantial computational resources, which leads to centralization, as only those with powerful mining hardware can compete effectively [6].

To address these issues, proof of stake (PoS) was introduced, which allocates block creation rights based on participants' stake percentages [7]. PoS introduces challenges like the 'Nothing at Stake' problem, where validators can vote for multiple chains without penalties. Additionally, those with larger stakes could dominate the network, undermining security and decentralization. DPoS improves scalability by allowing token holders to elect representatives to manage the consensus, but this can lead to power concentration, increasing the risk of network monopolies and vote manipulation [8].

Proof of elapsed time (PoET) [9] assigns block creation rights after a randomly assigned waiting period, which helps maintain network safety in low-power environments. However, the reliance on trusted execution environments (TEEs) introduces vulnerabilities, as TEE failures or security breaches could compromise the consensus process.

Proof of luck (PoL) [10] randomly assigns block creation opportunities to prevent centralization and promote fairness. However, its reliance on trusted execution environments (TEEs) limits participation to those with specific hardware, reducing diversity and making the network dependent on hardware configurations.

2.2. Verifiable Random Function

VRFs are cryptographic primitives that generate a random output along with a proof that the output was correctly computed [11]. This enables anyone to verify the correctness of the output without knowing the secret key. VRFs are characterized by uniqueness, pseudorandomness, and verifiability, making them particularly useful in blockchain protocols for ensuring fairness and security. In blockchain, VRFs are employed to enhance decentralization and security by preventing centralization and manipulation through verifiable randomness. For example, Algorand uses VRFs to randomly select block proposers and committee members, ensuring a fair and unbiased selection process.

However, the advent of quantum computing poses a significant threat to the security of existing VRF implementations that rely on classical cryptographic assumptions. Quantum computers threaten classical VRFs because they can solve problems like discrete logarithms much faster than traditional computers. This could break the cryptographic assumptions VRFs rely on, leaving blockchain systems vulnerable to attacks. To address this, new quantum-resistant VRF constructions are being developed.

2.2.1. Indexed Verifiable Random Function (iVRF)

The indexed verifiable random function (iVRF), introduced by Esgin, is a quantum-resistant extension of traditional VRFs tailored for blockchain leader election and consensus protocols [12]; iVRFs add crucial features like forward security, ensuring that past outputs remain secure even if the secret key is compromised. This is achieved using cryptographic hash functions, making iVRFs both scalable and secure for blockchain applications.

Traditional VRFs can encounter issues related to key refreshing, especially in blockchain environments, where long-term security is paramount. When using lattice-based post-quantum cryptographic schemes, more frequent key updates may be necessary to maintain security levels; iVRFs solve the issue of frequent key refreshes by using lattice-based hierarchical identity-based encryption (HIBE) schemes. This allows keys to be refreshed efficiently without replacing the entire key set, which is critical for long-term security in blockchain environments. In this system, a master secret key is used to derive round keys, which are each assigned a different identity (ID) and discarded after use. By employing this mechanism, iVRFs effectively manage the challenges associated with frequent key renewals, enhancing the overall security and efficiency of blockchain applications, where such functionality is critical. This means that even if the current secret key is compromised, all previous outputs remain secure, thereby maintaining the long-term security and integrity of the blockchain system.

In protocols like Algorand, iVRFs facilitate the random selection of block proposers and committee members while maintaining fairness and decentralization. The implementation of iVRFs demonstrates significant performance improvements over other quantum-safe VRF proposals, with faster evaluation and verification times; iVRFs strike a balance between proof size, computational efficiency, and communication overhead when compared to other post-quantum VRF models like lattice-based VRFs (LaVs) and ring-LWE-based VRFs [13,14].

2.2.2. Quantum Threats to Blockchain Security

In 1994, Peter Shor proposed a groundbreaking algorithm that could be executed on a quantum computer and capable of solving problems such as integer factorization and discrete logarithms, which are practically unsolvable by classical computers within a reasonable time, in polynomial time. The security of elliptic-curve cryptography (ECC) is based on the elliptic-curve discrete logarithm problem (ECDLP). However, quantum computers utilizing Shor's algorithm have the potential to break the security of ECC, meaning that information encrypted using ECC could be easily decrypted with the advent of quantum computers.

The security of blockchain primarily relies on maintaining the integrity and trustworthiness of transactions, with each transaction protected by cryptographic signatures verified through consensus among nodes. However, if a quantum computer can decrypt ECC signatures, an attacker could forge or alter existing blockchain transactions, severely undermining the fundamental trustworthiness of the blockchain. This could pose a significant threat to the network's security and integrity, particularly if quantum computers become commercially viable. Existing ECC-based blockchain systems would no longer be secure, risking the collapse of the entire blockchain structure [15,16]. Recent studies have explored quantum computing's impact on blockchain security and consensus mechanisms. The work in [17] demonstrates the potential profitability of quantum PoW mining. In [16], pre- and post-quantum cryptographic methods are compared, highlighting their resistance to quantum attacks. Ref. [18] reveals Bitcoin's vulnerability to Grover's algorithm, and [19] proposes a quantum-based identity authentication framework. The implementation of post-quantum blockchain systems faces significant challenges, particularly with respect to large ciphertext overheads and the compatibility of these schemes with existing blockchain infrastructures. Further research is required to optimize these cryptosystems and to standardize post-quantum cryptographic algorithms that can be seamlessly integrated into current blockchain networks.

Therefore, the advancement of quantum computing presents a significant challenge to blockchain technology. To maintain the security of blockchain, it is essential to adopt new post-quantum cryptography (PQC) algorithms that can replace existing cryptographic methods like ECC.

2.3. NIST Post-Quantum Cryptography

The NIST post-quantum cryptography (PQC) contest is an initiative to standardize cryptographic algorithms resilient to quantum attacks [20]. In the final round, the three candidate signature algorithms are CRYSTALS–DILITHIUM, FALCON, and SPHINCS+. Each of these algorithms has its own strengths and weaknesses, which are compared in Table 1.

Dilithium [21] is an algorithm based on the hardness of lattice problems and provides robust security and relatively efficient performance. It offers a balance between security and performance through various parameter settings but is limited by large signature sizes and a complex implementation.

Falcon [22] is a lattice-based algorithm utilizing FFTs for fast polynomial operations and boasts relatively small signature sizes and high security. However, it is complex to implement and can be resource-intensive on low-power devices. Despite being newer than Dilithium, it presents strong competitiveness in terms of efficiency and security.

SPHINCS+ [23] is a hash-based signature algorithm that provides strong resistance to quantum computer attacks. Based on simple cryptographic primitives, it ensures long-term security but suffers from very large signature sizes and slower signature generation and verification speeds. Although resource-intensive, its simple structure based on hash functions makes it relatively easy to implement.

Table 1. NIST PQC algorithm parameter sets and sizes.

Algorithm	**Parameter Set**	**NIST Security Level**	**Public Key Size (Bytes)**	**Secret Key Size (Bytes)**	**Signature Size (Bytes)**
Dilithium	Dilithium2	1	1312	2528	2420
	Dilithium3	3	1952	4000	3293
	Dilithium5	5	2592	4864	4459
Falcon	Falcon-512	1	897	1280	666
	Falcon-1024	5	1793	2305	1280
Sphincs+	Sphincs+-128s	1	32	64	8064
	Sphincs+-128f	1	32	64	17,088
	Sphincs+-192s	3	48	96	17,856
	Sphincs+-192f	3	48	96	35,664
	Sphincs+-256s	5	64	128	29,792
	Sphincs+-256f	5	64	128	49,856

3. PQ-DPoL

PQ-DPoL is a consensus algorithm designed to address quantum computing challenges while enhancing blockchain performance and security. This section provides an overview of the key components of PQ-DPoL and how they work together to create a resilient blockchain solution.

1. **Quantum resistance:** Traditional cryptographic methods are vulnerable to quantum computers, which can efficiently solve problems that are currently intractable for classical computers. To ensure long-term security against quantum threats, PQ-DPoL integrates post-quantum cryptography (PQC) algorithms such as the Falcon signature scheme, which is known for its small signature size and computational efficiency, making it well-suited for blockchain applications. This integration helps protect the blockchain system from the potential threats posed by quantum computing.
2. **Randomness and fairness:** The selection of nodes for block generation should be random and fair to prevent any single entity from dominating the network. PQ-DPoL leverages verifiable random functions (VRFs) to ensure that the selection process is both random and verifiable, thereby enhancing the fairness and security of the consensus mechanism. Randomness is crucial because it prevents predictable patterns in node selection, which could otherwise be exploited by attackers aiming to manipulate the network. By ensuring that the selection process is verifiable and random, PQ-DPoL enhances decentralization and overall network security.
3. **Execution speed:** Achieving consensus in a distributed network should be computationally efficient to minimize processing time. PQ-DPoL optimizes execution speed by reducing computational overhead and streamlining the consensus process. Fast

transaction finality is essential for maintaining blockchain performance, particularly for block generation and validation.

4. **Energy efficiency:** Minimizing energy consumption is crucial for the sustainability of blockchain networks. PQ-DPoL uses a delegated approach to reduce the number of nodes involved in the consensus process at any given time, significantly cutting down on energy consumption compared to traditional methods like proof of work (PoW).

PQ-DPoL addresses key blockchain challenges by combining quantum resistance, randomness, fairness, execution speed, and energy efficiency. These elements work together to create a scalable and robust blockchain solution that is designed to enhance both security and performance.

Figure 1 illustrates how PQ-DPoL operates in rounds, with each consisting of two main stages: the selection of delegate nodes and block generation and verification. At the beginning of each round, nodes generate random values using an indexed verifiable random function (iVRF). The nodes with the smallest random values are chosen as delegate nodes, as this method ensures an unbiased selection process that is both fair and transparent.

Once the delegate nodes are selected, they proceed with block generation and verification using the practical Byzantine fault tolerance (PBFT) method. The primary delegate node proposes a block, and the other delegate nodes verify it. If validated, the block is added to the blockchain, ensuring secure and efficient block generation and verification.

After the block is added to the blockchain, the round is complete, and the process begins anew with the selection of new delegate nodes. This cyclical process ensures continuous block creation while maintaining security and fairness within the network.

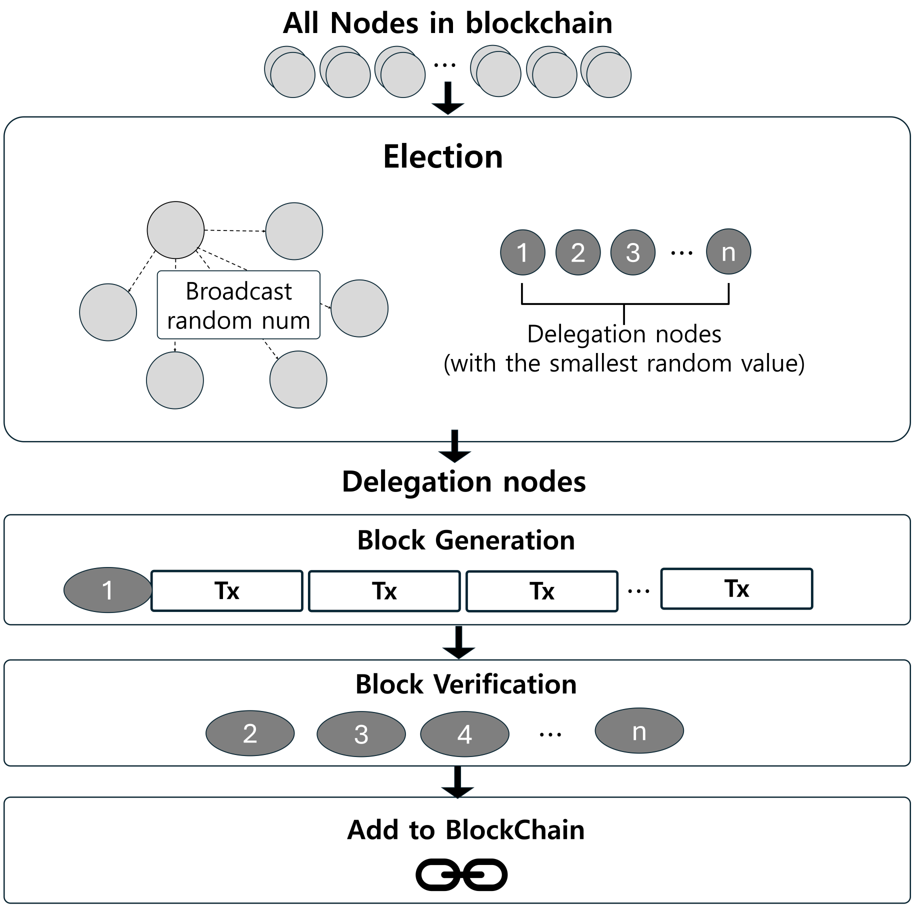

Figure 1. Overview of PQ-DPoL.

3.1. Delegation

To improve efficiency, DPoL delegates block generation and verification to a smaller number of nodes. The delegation process improves efficiency by assigning block generation and verification to a small subset of delegate nodes (N nodes), reducing the overall workload and speeding up the consensus process.

At the start of the consensus process, each node generates a random value to select delegate nodes. This ensures that the selection is fair, unpredictable, and secure, preventing any entity from manipulating the process. Each node uses an indexed verifiable random function (iVRF) to generate a random value based on the hash of the previous block and the node's ID. This value is then broadcast to the network for verification by other nodes. This process ensures the integrity of the random values and prevents any manipulation or malicious activity, maintaining the security and reliability of the network.

The iVRF ensures that the selection of delegate nodes remains unpredictable and verifiable, enhancing security while maintaining a decentralized network structure. By using an iVRF, the system maintains decentralization and prevents any single entity from gaining disproportionate control, thereby safeguarding the integrity of the blockchain. Additionally, the iVRF's key refresh mechanism imposes minimal overhead, ensuring the network maintains high security standards without sacrificing performance.

Fairness through Random Selection

To ensure fairness in node selection, each node generates a random value using the iVRF, which ensures unpredictability and security in the random value generation process. Moreover, the iVRF leverages lattice-based hierarchical identity-based encryption (HIBE) schemes to incorporate forward security. This approach allows for efficient key refreshing without the need to frequently replace the entire key set, addressing potential issues with key updates in blockchain systems. Once all nodes have generated their random values and transmitted them to other nodes, each node verifies the received random values. After verification, the nodes compare the verified random values and select the N nodes with the smallest values as delegate nodes for the next phase. Selecting nodes based on the smallest random values ensures that the process is fair and unpredictable. This minimizes the risk of manipulation or collusion, preventing any entity from gaining an unfair advantage and maintaining the integrity of the network. This approach helps prevent centralization and ensures that every node has an equal probability of being selected as a delegate, thereby maintaining the decentralization and security of the blockchain.

As described in Algorithm 1, the process of generating and broadcasting iVRF values begins with each node using its secret key (SK_A), a specific input, and the hash of the previous block (H_{prev}) to generate a random value (R_A) and a corresponding proof (π_A) through an indexed verifiable random function (iVRF). The function iVRF_GenerateAndBroadcast takes these inputs and produces the random value and proof pair. Once generated, the node broadcasts the pair (R_A, π_A) to the network, allowing other nodes to verify the correctness of the random value using the proof.

Algorithm 1 Generating and broadcasting iVRF values

```
Input: $input$, $SK_A$ (secret key of Node A), $H_{prev}$ (previous block hash)
Output: $R_A$ (random value), $\pi_A$ (proof)
function iVRF_GenerateAndBroadcast($input$, $SK_A$, $H_{prev}$)
    $(R_A, \pi_A) \leftarrow$ iVRF$(SK_A, (input, H_{prev}))$
    Broadcast $(R_A, \pi_A)$ to the network
end function
```

In Algorithm 2, the iVRF (indexed verifiable random function) verification process requires each node to validate the random values generated by others. During the verification, iVRF values are compared through Algorithm 3. As the network expands, this process scales with the number of nodes, potentially increasing the computational load. PQ-DPoL mitigates this computational load by selecting a small subset of nodes, called delegate nodes, to handle block generation and verification. Selecting delegate nodes is performed in Algorithm 4. This significantly reduces the overall processing demand on the network, even as the number of nodes grows. This delegation ensures that the performance impact remains low even as the network grows. While iVRF involves an initial key exchange during network setup, this is a one-time overhead that does not affect regular operations. Once the initial exchange is completed, it does not introduce further delays during block generation or verification.

Algorithm 2 Verifying iVRF values

```
1: **Input:** $(R_A, \pi_A)$ from Node A, $PK_A$ (public key of Node A)
2: **Output:** Validity of $(R_A, \pi_A)$
3: **for** each $(R_i, \pi_i)$ received from node $i$ **do**
4:     **if** Verify$(PK_i, input, R_i, \pi_i) \neq$ true **then**
5:         Reject $(R_i, \pi_i)$
6:     **end if**
7: **end for**
```

Algorithm 3 Comparing iVRF values

```
1: **Input:** $H_{prev}$ (previous block hash), $(R_i, \pi_i)$ (verified iVRF values from all nodes)
2: **Output:** $NodeScores$
3: $NodeScores \leftarrow$ empty list
4: **for** each $R_i$ in the set of verified iVRF values **do**
5:     $score_i \leftarrow$ abs$(H_{prev} - R_i)$
6:     Append $(score_i, node_i)$ to $NodeScores$
7: **end for**
```

The nodes with scores closest to the previous block's hash are selected as delegate nodes, which will be responsible for generating and validating blocks in the next phase.

Algorithm 4 Selecting delegate nodes

```
1: **Input:** $NodeScores$
2: **Output:** $SelectedNodes$
3: Sort $NodeScores$ by $score$ in ascending order
4: $SelectedNodes \leftarrow$ first $n$ nodes from $NodeScores$
5: **return** $SelectedNodes$
```

3.2. Block Generation and Verification

As described in Algorithm 5, the selected nodes use the practical Byzantine fault tolerance (PBFT) [24] method to generate and verify blocks. PBFT ensures that all honest nodes agree on the sequence of blocks, maintaining the consistency and integrity of the blockchain, even in the presence of malicious nodes. The PBFT process is divided into four main phases:

Pre-Preparation phase: the primary node proposes a block, including a sequence number.

Preparation phase: replica nodes verify the block and send prepare messages to confirm its validity.

Commitment phase: after receiving prepare messages from two-thirds of the nodes, replica nodes send commit messages.

Finalization phase: once commit messages from two-thirds of the nodes are collected, the block is added to the blockchain, and a confirmation message is broadcast.

In PQ-DPoL, PBFT ensures both security and efficiency by avoiding energy-intensive computations, making it a suitable consensus mechanism for a post-quantum environment. By integrating PBFT, PQ-DPoL is resilient against quantum threats while maintaining high throughput and scalability. To address the challenges posed by post-quantum cryptography (PQC), such as larger keys and signatures, PQ-DPoL incorporates a block compression algorithm. This ensures faster processing and storage efficiency, even with the larger cryptographic parameters introduced by PQC. Experimental results demonstrate that the combination of PBFT and PQC within PQ-DPoL enhances both network security and performance, ensuring scalability and quantum resistance while supporting a high number of transactions per second (TPSs).

Algorithm 5 Block generation and verification

```
1:  Input: $ProposedBlock$, $SequenceNumber$
2:  Output: $FinalizedBlock$
3:  Phase 1: Pre-Preparation
4:  Primary node $P$ proposes $ProposedBlock$ with $SequenceNumber$
5:  $P$ sends $(ProposedBlock, SequenceNumber)$ to all replicas $R_i$
6:  Phase 2: Preparation
7:  for each replica node $R_i$ do
8:      Verify $ProposedBlock$
9:      if $ProposedBlock$ is valid then
10:         Send $Prepare(SequenceNumber, H(ProposedBlock))$ to all replicas $R_j$
11:     end if
12:     Collect $Prepare$ messages from other replicas $R_j$
13: end for
14: Phase 3: Commitment
15: for each replica node $R_i$ do
16:     if received $Prepare$ messages from $\geq \frac{2n}{3}$ replicas then
17:         Enter the Commitment phase
18:         Send $Commit(SequenceNumber, H(ProposedBlock))$ to all replicas $R_j$
19:     end if
20:     Collect $Commit$ messages from other replicas $R_j$
21: end for
22: Phase 4: Finalization
23: for each replica node $R_i$ do
24:     if received $Commit$ messages from $\geq \frac{2n}{3}$ replicas then
25:         Add $ProposedBlock$ to blockchain
26:         Broadcast confirmation message to network
27:     end if
28: end for
```

3.3. Block Compression

In blockchain systems, replacing ECC (elliptic-curve cryptography) with PQC (post-quantum cryptography) introduces significant challenges due to the increased size of signatures and public keys. As shown in Table 2, NIST PQC algorithms like Falcon and Dilithium offer relatively smaller public key and signature sizes compared to other PQC options, such as SPHINCS+, which has significantly larger signatures. However, even with Falcon and Dilithium, the increase in key and signature sizes can substantially reduce the number of transactions that can be stored within a fixed-size block, thereby decreasing the transactions per second (TPSs) by more than 10×. Although Falcon and Dilithium present a more balanced trade-off between security and performance, their adoption still necessitates careful consideration of the impact on blockchain scalability and efficiency. The use of these algorithms helps to mitigate some of the performance overhead associated with larger cryptographic parameters, but the reduction in TPSs remains a significant challenge for maintaining the performance and scalability of blockchain networks.

To address this reduction in TPSs and ensure the blockchain network remains efficient, block compression is employed. Compression mitigates the impact of the larger key and signature sizes introduced by PQC, allowing the network to maintain high TPSs.

Block compression helps PQ-DPoL mitigate performance bottlenecks caused by large post-quantum signatures, allowing the system to maintain scalability and efficiency.

For block compression, Snappy, a fast compression algorithm commonly used in real-time data processing, is utilized [25]. Snappy, known for its balance between compression speed and efficiency, is chosen to handle the larger data sizes introduced by PQC. This ensures that PQ-DPoL can maintain high throughput and performance by compressing data before transmission, minimizing the impact of larger signatures and keys without compromising security or integrity. Its rapid compression capabilities ensure that even with the larger sizes introduced by PQC, the network can maintain high throughput. This compression step enables faster transmission and processing while maintaining the security and integrity of transactions. However, while the chosen Snappy compression algorithm provides fast processing speeds, its compression ratio may be lower compared to other high-compression algorithms. This could potentially result in insufficient reduction of data size, leading to limitations in storage efficiency within the blockchain network. To assess this, we analyze the compression efficiency of the Snappy algorithm when applied to blocks in Section 4.2.

Table 2. Comparison of ECC, Falcon, Dilithium, and SPHINCS+ key and signature sizes (bytes).

Algorithm	Public Key Size	Private Key Size	Signature Size	Public Key + Signature
ECC 256	64	32	64	128
Falcon-512	897	1280	666	1563
Dilithium2	1312	2528	2420	3732
SPHINCS+-SHA2-128s	32	64	7856	7888
SPHINCS+-SHAKE-128f	32	64	17,088	17,120
ECC 384	96	48	96	192
Falcon-1024	1793	2304	1280	3073
Dilithium3	1952	4000	3293	5245
SPHINCS+-SHA2-192f	48	96	35,664	35,712
SPHINCS+-SHA2-192s	48	96	16,224	16,272
Dilithium5	2592	4864	4595	7187
SPHINCS+-SHA2-256f	64	128	49,856	49,920
SPHINCS+-SHA2-256s	64	128	29,792	29,856

4. Experiments

The experiments were conducted using C++ on a system equipped with an Intel i5-8295U CPU and 16 GB RAM and running Ubuntu 20.04.6 LTS. For the post-quantum signature algorithms, Falcon and Dilithium, we utilized the QS-OpenSSL-1.1.1 library, which supports quantum-safe cryptographic algorithms. This section presents a detailed performance analysis comparing the traditional elliptic-curve digital signature algorithm (ECDSA), the Falcon post-quantum signature algorithm, and the Dilithium post-quantum signature algorithm. Key metrics evaluated include the number of signatures generated per second (Sign/s) and the number of verifications per second (Verify/s), as well as the impact on transactions per second (TPSs) for blockchain operations.

4.1. Signature Algorithm Comparison

In this subsection, we present a comprehensive performance comparison between the elliptic-curve digital signature algorithm (ECDSA), the Falcon post-quantum signature algorithm, and the Dilithium post-quantum signature algorithm by using the 'openssl speed' command for measurement. The Falcon and Dilithium algorithms were tested using the QS-OpenSSL-1.1.1 library, which provides implementations of these quantum-safe algorithms. The metrics evaluated include the number of signatures generated per second (Sign/s) and the number of verifications per second (Verify/s).

Table 3 highlights the performance differences between the ECDSA, Falcon algorithm, and Dilithium algorithm. Notably, both Falcon and Dilithium significantly outperform ECDSA in verification speed. Specifically, Dilithium2 achieves the highest verification speed of 41,114.4 verifications per second, followed by Falcon-512 with 24,078.9 verifications per second, compared to 16,063.7 verifications per second for ECC 256. This increased efficiency in verification makes both Falcon and Dilithium highly compelling choices for blockchain environments, where quick transaction validation is essential.

Table 3. Performance comparison of ECDSA, Falcon, and Dilithium for signing and verification.

Algorithm	Sign/s	Verify/s
ECC 256	46,900.2	16,063.7
ECC 384	1106.3	1370.7
Falcon-512	4107.6	24,078.9
Falcon-1024	2046.4	11,867.8
Dilithium2	15,197.0	41,114.4
Dilithium5	7583.7	15,479.5

However, while Falcon offers robust post-quantum security, Dilithium has certain advantages in both signing and verification speeds. For instance, Dilithium2 generates 15,197.0 signatures per second, outperforming both Falcon-512 (4107.6 signatures per second) and Falcon-1024 (2046.4 signatures per second). This makes Dilithium2 a more efficient choice in environments where higher signing speeds are required. Similarly, Dilithium5 performs better in both signing and verification than Falcon-1024, generating 7583.7 signatures per second and verifying 15,479.5 verifications per second.

One key trade-off between Falcon, Dilithium, and ECDSA lies in the increased signature and public key sizes of the post-quantum cryptographic algorithms. Larger signature sizes in Falcon and Dilithium require more storage space, potentially reducing the number of transactions processed per block, which can negatively impact the transaction per second (TPS) rate of the blockchain.

4.2. Compression Efficiency Analysis

In order to address the increased signature sizes associated with post-quantum cryptography algorithms such as Falcon and Dilithium, we incorporated a block compression algorithm to optimize the storage and transmission efficiency. The graph in Figure 2 illustrates the compression ratio achieved as the size of the transaction data increases. The compression algorithm was tested with varying data sizes ranging from 10,000 to 100,000 transactions. The corresponding compression ratios (%) were measured. As seen in Figure 2, the compression ratio increases as the data size grows, reaching a peak of 42.12% at 80,000 transactions. Beyond this point, the compression ratio stabilizes between 40% and 42%. This suggests that the compression algorithm is particularly effective for larger data sets, reducing storage requirements by over 40% in scenarios with large transaction volumes. This improvement in data compression allows for better scalability of the blockchain network by accommodating the larger signature sizes introduced by post-quantum algorithms. Moreover, the compression helps to offset the increase in storage space required for post-quantum signatures, ultimately enhancing the blockchain's efficiency without compromising security.

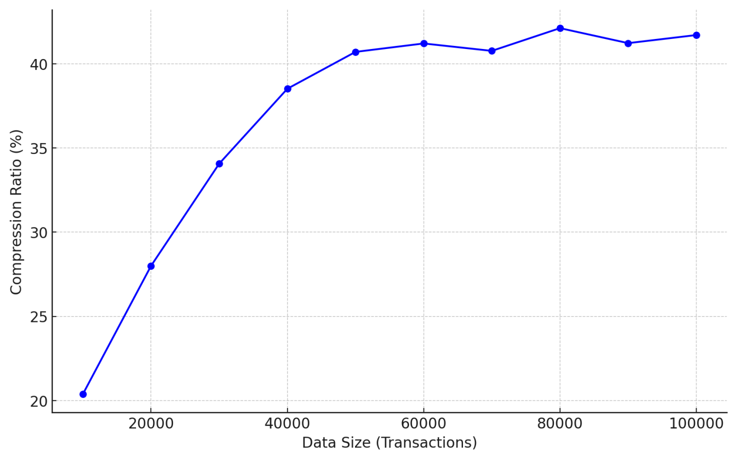

Figure 2. Compression ratio improvement as a function of increasing transaction data size.

4.3. Compression TPSs

To simulate a realistic blockchain network environment, the open-source network simulator NS-3 was utilized. The performance and scalability of the blockchain network were evaluated based on the transactions per second (TPSs) metric. All related data and codes used in this simulation are publicly available at (https://github.com/amdjd0704/pq-dpol (accessed on 12 September 2024)). We utilized the NS3 simulator to implement a fully meshed network topology. The network consisted of N nodes, where each node was directly connected to every other node using point-to-point links. The data rate for each link was set to 3 Mbps with a latency of 3 ms. Each node was assigned an IP address within the 1.0.0.0/24 subnet, and the Internet stack was installed on all nodes. The simulation was configured using NS3's InternetStackHelper and PointToPointHelper modules, and the fully meshed network topology allowed for performance analysis of message transmission between nodes.

Table 4 shows that the TPS measurements for the Falcon and ECC algorithms reveal significant differences in their performance. Falcon demonstrates lower TPS values compared to ECC across various transaction counts, reflecting the impact of larger signature sizes and more complex cryptographic operations. For instance, with a transaction count of 3, Falcon achieves a TPS count of 62.86, which is significantly lower than ECC's TPS count of 118.25, indicating that Falcon's performance is roughly 53% of ECC's at this transaction count. This trend continues as the transaction count increases. At a transaction count of 10, Falcon's TPS count is 111.11, while ECC's TPS count is 349.31, showing that Falcon performs at about 31% of ECC's efficiency. The gap remains consistent across higher transaction counts, with Falcon reaching a maximum TPSs of 136.36 at a transaction count of 18, whereas ECC reaches 563.66 at a transaction count of 19, meaning Falcon's TPSs are approximately 24% of ECC's.

While Falcon offers robust security against quantum threats, its larger cryptographic keys and signatures slightly reduce TPSs. However, the performance remains competitive despite these more complex and data-intensive operations, making Falcon viable for blockchain systems prioritizing long-term security. This makes Falcon a viable option for blockchain systems that prioritize long-term security over immediate throughput.

To further understand these performance implications, it is important to consider the underlying reasons for the observed differences. The larger size of Falcon's cryptographic keys and signatures necessitates more computational resources for processing each transaction. This increased computational load translates to longer processing times and higher data storage requirements, which directly impact the TPSs. However, Falcon's advantage lies in its enhanced security measures, which are designed to withstand quantum attacks, thereby providing a higher level of security assurance for blockchain applications.

Table 4. Transaction count and TPS data. (a) TPSs for different block sizes in Falcon. (b) TPSs for different block sizes in ECC.

(a)		**(b)**	
Transaction Count	**TPSs**	**Transaction Count**	**TPSs**
3	62.864569	3	118.256962
4	73.939394	4	154.752654
5	84.183673	5	189.078732
6	89.090909	6	223.209768
7	94.040404	7	255.014327
8	106.122449	8	286.120760
9	110.909091	9	316.528807
10	111.111111	10	349.315068
11	115.555556	11	373.404255
12	126.060606	12	399.478465
13	123.434343	13	425.352372
14	125.858586	14	451.963348
15	139.393939	15	474.535315
16	133.877551	16	497.561441
17	135.656566	17	521.004808
18	136.363636	18	542.065868
19	136.262626	19	563.662381

We further explore the impact of various post-quantum cryptographic algorithms on transactions per second (TPSs). To provide a comprehensive comparison, we measured the TPSs across multiple transaction counts and sizes for both the Falcon and Dilithium algorithms. Figure 3 presents a visual representation of the results, showing the TPS performances for Falcon-512, Falcon-1024, Dilithium-2, and Dilithium-5.

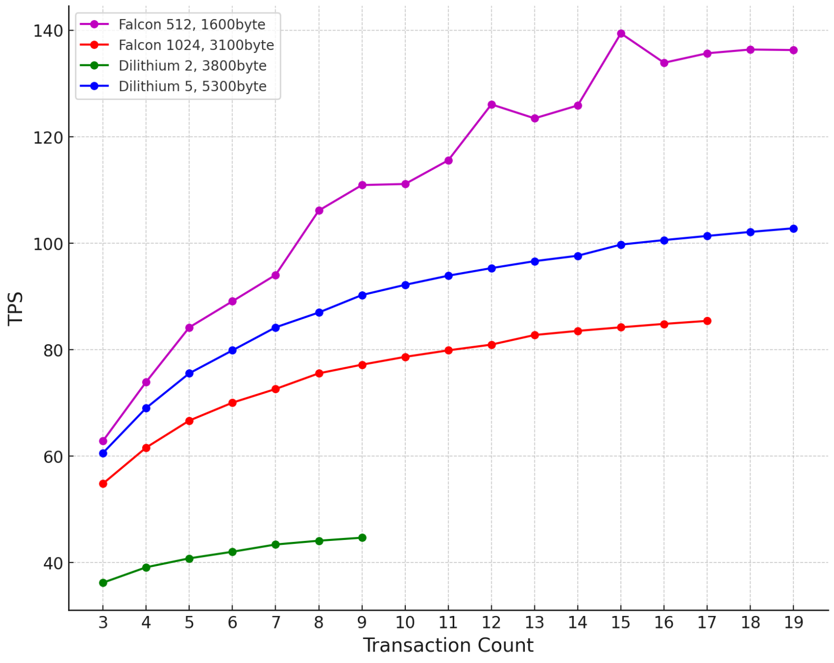

Figure 3. TPS comparison for different transaction counts and sizes using Falcon and Dilithium post-quantum cryptographic algorithms.

Falcon-512 shows strong post-quantum security while maintaining a peak TPS of 136 at a transaction count of 18, making it well-suited for environments demanding both security and performance. Falcon-1024, while also delivering robust post-quantum security, shows lower TPS values compared to Falcon-512 due to its larger transaction size (3100 bytes).

However, its performance remains competitive, reaching a TPS count of approximately 64 at a transaction count of 19. This demonstrates that Falcon-1024 can still be a viable option for systems that require even stronger security assurances, albeit with a moderate reduction in TPS. On the other hand, the Dilithium algorithms exhibit lower TPS values across all transaction counts. Dilithium-2, which has a transaction size of 3800 bytes, achieves a maximum TPS value of only 44, and Dilithium-5, despite offering stronger security with a transaction size of 5300 bytes, peaks at around 84 TPS. The larger transaction sizes of the Dilithium algorithms, while providing enhanced security, result in a significant reduction in TPSs, making them less efficient for high-throughput blockchain systems.

However, it is important to note that these measurements were obtained in a simulated environment using NS-3. Due to the constraints of the simulation environment, only small block sizes could be tested, which limits the extent to which the results can represent real-world performance. Especially in large-scale blockchain networks, block sizes may vary significantly, and the results from this experiment are derived from smaller-scale testing. Larger block sizes in actual implementations may yield different performance metrics. By quantifying these differences, it becomes evident that Falcon ensures robust security against potential quantum threats at the cost of performance efficiency. Despite the larger size of Falcon's cryptographic keys and signatures, the reduction in transactions per second (TPSs) is moderate. Falcon's cryptographic operations are more complex and data-intensive, yet its performance remains competitively efficient. This makes Falcon a viable option for blockchain systems that prioritize long-term security over immediate throughput. These results underscore the trade-offs between security and performance in the context of post-quantum cryptography. While Dilithium provides strong quantum resistance, its larger transaction sizes introduce substantial overhead, reducing the network's throughput. In contrast, Falcon, particularly Falcon-512, strikes a more favorable balance between security and performance, maintaining a higher TPS value even with post-quantum cryptographic requirements. This makes Falcon a more suitable choice for blockchain applications, where transaction throughput is critical, without sacrificing the necessary level of post-quantum security.

5. Conclusions

This paper introduced the post-quantum delegated proof of luck (PQ-DPoL): a blockchain consensus algorithm designed to counter the growing threat posed by quantum computing. By integrating the Falcon and Dilithium post-quantum signature algorithms within a delegated consensus mechanism, PQ-DPoL strikes a balance between security, performance, and fairness. The system is designed to ensure resilience against quantum attacks, safeguarding blockchain networks as quantum computing advances.

Our experiments highlight the trade-offs associated with post-quantum cryptography, particularly in terms of transactions per second (TPSs). While Falcon provides robust quantum resistance, its larger signatures reduce TPS values compared to traditional ECC-based solutions. To mitigate this, PQ-DPoL employs efficient compression algorithms, reducing block sizes without compromising security. Dilithium, with faster signing and verification times, also presents performance benefits but suffers from larger transaction sizes, impacting TPS values in high-throughput environments.

The experiments were conducted in a simulated environment with certain limitations such as smaller block sizes and computational constraints that restrict testing at larger-scale, real-world scenarios. Consequently, the current TPS measurements may not fully represent PQ-DPoL's performance under larger block sizes or higher node counts, emphasizing the need for further exploration in more extensive environments. Despite these limitations, Falcon demonstrated competitive performance, making it a viable candidate for blockchain systems prioritizing long-term security.

Future work will focus on overcoming the limitations of the simulation environment, particularly by testing PQ-DPoL in larger-scale networks and with varying block sizes to assess its scalability and efficiency. Additionally, further research will explore optimizing compression algorithms to improve TPS values without compromising security. Investigating other PQC algorithms with improved performance characteristics and refining the data compression techniques in PQ-DPoL will also be key areas of future exploration.

Author Contributions: Software, H.K. (Hyunji Kim); Investigation, W.K. and Y.K.; Writing—original draft, H.K. (Hyunjun Kim) and H.K. (Hyunji Kim); Writing—review & editing, H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00264, Research on Blockchain Security Technology for IoT Services, 50%) and this work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00627, Development of Lightweight BIoT technology for Highly Constrained Devices, 50%).

Data Availability Statement: Source codes are available in https://github.com/amdjd0704/pq-dpol (accessed on 12 September 2024).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sharma, P.; Jindal, R.; Borah, M.D. A review of blockchain-based applications and challenges. *Wirel. Pers. Commun.* **2022**, *123*, 1–43. [CrossRef]
2. Panda, S.K.; Mishra, V.; Dash, S.P.; Pani, A.K. *Recent Advances in Blockchain Technology: Real-World Applications*; Springer: Cham, Switzerland, 2023.
3. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **1999**, *41*, 303–332. [CrossRef]
4. Vidaković, M.; Miličević, K. Performance and Applicability of Post-Quantum Digital Signature Algorithms in Resource-Constrained Environments. *Algorithms* **2023**, *16*, 518. [CrossRef]
5. Kim, W.; Kang, Y.; Kim, H.; Jang, K.; Seo, H. PQ-DPoL: An Efficient Post-Quantum Blockchain Consensus Algorithm. In Proceedings of the International Conference on Information Security Applications, Jeju Island, Republic of Korea, 23 August 2023; Springer: Berline/Heidelberg, Germany 2023; pp. 310–323.
6. Gervais, A.; Karame, G.O.; Wüst, K.; Glykantzis, V.; Ritzdorf, H.; Capkun, S. On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 3–16.
7. Nguyen, C.T.; Hoang, D.T.; Nguyen, D.N.; Niyato, D.; Nguyen, H.T.; Dutkiewicz, E. Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities. *IEEE Access* **2019**, *7*, 85727–85745. [CrossRef]
8. Saad, S.M.S.; Radzi, R.Z.R.M. Comparative review of the blockchain consensus algorithm between proof of stake (pos) and delegated proof of stake (dpos). *Int. J. Innov. Comput.* **2020**, *10*.
9. Chen, L.; Xu, L.; Shah, N.; Gao, Z.; Lu, Y.; Shi, W. On security analysis of proof-of-elapsed-time (poet). In Proceedings of the Stabilization, Safety, and Security of Distributed Systems: 19th International Symposium, SSS 2017, Boston, MA, USA, 5–8 November 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 282–297.
10. Milutinovic, M.; He, W.; Wu, H.; Kanwal, M. Proof of luck: An efficient blockchain consensus protocol. In Proceedings of the 1st Workshop on System Software for Trusted Execution, Trento, Italy, 12–16 December 2016; pp. 1–6.
11. Micali, S.; Rabin, M.; Vadhan, S. Verifiable random functions. In Proceedings of the 40th Annual Symposium on Foundations of Computer Science (cat. No. 99CB37039), New York, NY, USA, 17–19 October 1999; pp. 120–130.
12. Esgin, M.F.; Ersoy, O.; Kuchta, V.; Loss, J.; Sakzad, A.; Steinfeld, R.; Yang, X.; Zhao, R.K. A new look at blockchain leader election: Simple, efficient, sustainable and post-quantum. In Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security, Melbourned, VIC, Australia, 10–14 July 2023; pp. 623–637.
13. Esgin, M.F.; Steinfeld, R.; Liu, D.; Ruj, S. Efficient hybrid exact/relaxed lattice proofs and applications to rounding and VRFs. In Proceedings of the Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, 20 August 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 484–517.
14. Kim, B.G.; Wong, D.; Yang, Y.S. Private and secure post-quantum verifiable random function with nizk proof and ring-lwe encryption in blockchain. *arXiv* **2023**, arXiv:2311.11734.
15. Allende, M.; León, D.L.; Cerón, S.; Pareja, A.; Pacheco, E.; Leal, A.; Da Silva, M.; Pardo, A.; Jones, D.; Worrall, D.J.; et al. Quantum-resistance in blockchain networks. *Sci. Rep.* **2023**, *13*, 5664. [CrossRef] [PubMed]
16. Fernandez-Carames, T.M.; Fraga-Lamas, P. Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks. *IEEE Access* **2020**, *8*, 21091–21116. [CrossRef]
17. Bard, D.A.; Kearney, J.J.; Perez-Delgado, C.A. Quantum advantage on proof of work. *Array* **2022**, *15*, 100225. [CrossRef]

18. Holmes, S.; Chen, L. Assessment of quantum threat to bitcoin and derived cryptocurrencies. *Cryptol. Eprint Arch.* Available online: https://eprint.iacr.org/2021/967 (accessed on 12 September 2024).
19. Yang, Z.; Salman, T.; Jain, R.; Di Pietro, R. Decentralization using quantum blockchain: A theoretical analysis. *IEEE Trans. Quantum Eng.* **2022**, *3*, 4100716. [CrossRef]
20. Dam, D.T.; Tran, T.H.; Hoang, V.P.; Pham, C.K.; Hoang, T.T. A survey of post-quantum cryptography: Start of a new race. *Cryptography* **2023**, *7*, 40. [CrossRef]
21. Lyubashevsky, V.; Ducas, L.; Kiltz, E.; Lepoint, T.; Schwabe, P.; Seiler, G.; Stehlé, D.; Bai, S. Crystals-dilithium. In *Algorithm Specifications and Supporting Documentation*. Available online: https://csrc.nist.gov/CSRC/media/Presentations/crystals-dilithium-round-3-presentation/images-media/session-1-crystals-dilithium-lyubashevsky.pdf (accessed on 12 September 2024).
22. Prest, T.; Fouque, P.A.; Hoffstein, J.; Kirchner, P.; Lyubashevsky, V.; Pornin, T.; Ricosset, T.; Seiler, G.; Whyte, W.; Zhang, Z. Falcon. In *Post-Quantum Cryptography Project of NIST*. Available online: https://www.post-quantum.nl/slides/PQC4_Slides_Thomas.Prest.pdf (accessed on 12 September 2024).
23. Bernstein, D.J.; Hülsing, A.; Kölbl, S.; Niederhagen, R.; Rijneveld, J.; Schwabe, P. The SPHINCS+ signature framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 11–15 November 2019; pp. 2129–2146.
24. Castro, M.; Liskov, B. Practical byzantine fault tolerance. In Proceedings of the 3rd Symposium on Operating Systems Design and Implementation, OsDI, New Orleans, LA, USA, 22–25 February 1999; Volume 99, pp. 173–186.
25. Snappy. Available online: https://google.github.io/snappy/ (accessed on 22 July 2024).