Enhancing blockchain scalability through zero-knowledge proofs: A novel block finality system for near protocol

Oleksandr Kuznetsov[1,2,*,†], Anton Yezhov[2,†], Kateryna Kuznetsova[2,3,†], Vladyslav Yusiuk[2,†] and Valentyn Chernushevych[2,†]

[1] *eCampus University, 10 Via Isimbardi, 22060 Novedrate, Italy*
[2] *Zpoken OÜ, Harju maakond, Kesklinna linnaosa, 7-2 Sakala tn, 10141 Tallinn, Estonia*
[3] *Vision, Robotics and Artificial Intelligence Lab, 12 Via Brecce Bianche, 60131 Ancona, Italy*

Abstract
This paper presents a novel zero-knowledge proof (ZKP) system for block finality verification in the NEAR Protocol, addressing critical challenges in blockchain scalability and security. We introduce a comprehensive ZKP-based verification system that encompasses block hash, signature, validator key and stake, and next block producer hash verification. Our approach achieves constant-time verification for light clients, regardless of block size or complexity, significantly enhancing the efficiency and security of light client operations. By leveraging advanced cryptographic techniques, including the Plonky2 framework, we demonstrate the feasibility of using ZKPs in high-throughput blockchain networks. Our performance evaluation provides valuable insights into the scalability and efficiency of ZKP systems in real-world blockchain environments. Results show consistent proof verification times of approximately 4 milliseconds across varying block sizes, with aggregated proof sizes remaining constant at 180,112 bytes. While proof generation times range from 13 to 18 minutes per block, the rapid verification time and compact proof size offer substantial benefits for light clients and cross-chain communication. This work contributes to the ongoing research in blockchain scalability, offering a practical solution that maintains security and decentralization while significantly reducing computational and bandwidth requirements for blockchain participants.

Keywords
distributed system, light client, blockchain scalability, zero-knowledge proof, NEAR protocol, block finality verification, cybersecurity, blockchain interoperability

1. Introduction

Blockchain technology has emerged as a transformative force in various sectors, promising enhanced security, transparency, and decentralization [1]. However, as blockchain applications proliferate, scalability has become a critical challenge, particularly for high-throughput systems like the NEAR Protocol [2]. The increasing demand for faster transaction processing and more efficient data verification has pushed the limits of traditional blockchain architectures, necessitating innovative solutions to maintain the technology's core benefits while improving its scalability [3].

The NEAR Protocol, designed as a sharded, proof-of-stake blockchain, aims to address some of these scalability issues through its unique architecture [4]. However, as with many blockchain systems, the challenge of efficient block finality verification, especially for light clients, remains a significant hurdle [5, 6]. Light clients, crucial for broadening blockchain accessibility, often struggle with the trade-off between efficiency and security guarantees when verifying the blockchain state [7].

This paper introduces a novel zero-knowledge proof (ZKP) system designed specifically for block finality verification in the NEAR Protocol. Our approach leverages advanced cryptographic techniques to create a system that allows for rapid and secure verification of block finality, particularly beneficial for light clients and cross-chain communication scenarios.

The core contributions of this work include:

- A comprehensive ZKP-based verification system that encompasses block hash, signature, validator

CPITS-II 2024: Workshop on Cybersecurity Providing in Information and Telecommunication Systems II, October 26, 2024, Kyiv, Ukraine
* Corresponding author.
† These authors contributed equally.
oleksandr.kuznetsov@uniecampus.it, oleksandr.k@zpoken.io (O. Kuznetsov);
anton.yezhov@zpoken.io (A. Yezhov);
kateryna@proxima.one (K. Kuznetsova);
vladyslav.y@zpoken.io (V. Yusiuk);
valentine@zpoken.io (V. Chernushevych)

0000-0003-2331-6326 (O. Kuznetsov);
0009-0004-6380-5233 (A. Yezhov);
0000-0002-5605-9293 (K. Kuznetsova);
0009-0009-9662-9615 (V. Yusiuk);
0009-0007-4186-1981 (V. Chernushevych)

key and stake, and next block producer hash verification.

- Implementation of constant-time verification for light clients, independent of block size or complexity, significantly enhancing the efficiency and security of light client operations.
- Practical demonstration of the Plonky2 framework's capabilities in generating and verifying zero-knowledge proofs for high-throughput blockchain networks.
- Extensive performance evaluation providing insights into the scalability and efficiency of ZKP systems in real-world blockchain environments.
- Analysis of the trade-offs between proof generation time, verification speed, and proof size, offering valuable data for future blockchain design decisions.

Our work builds upon a growing body of research in blockchain scalability, zero-knowledge proofs, and distributed systems. By addressing the specific challenges of the NEAR Protocol while maintaining broader applicability, we contribute to the ongoing evolution of blockchain technology toward more scalable and efficient systems.

The remainder of this paper is structured as follows: Sections 2 and 3 provide a background on relevant concepts and related work. Section 4 details the architecture of our ZKP-based block finality system. Section 5 describes the zero-knowledge proof construction. Section 6 offers a comprehensive performance evaluation. Section 7 discusses the implications of our findings and potential future directions. Finally, Section 8 concludes the paper with a summary of our key contributions and their significance in the field of blockchain technology.

2. Literature review

The development of scalable and secure blockchain systems has been a focal point of research in recent years, driven by the need to address the limitations of early blockchain implementations. This section provides an overview of key contributions and ongoing challenges in this domain.

Scalability has emerged as a critical issue in blockchain systems, as highlighted by Nasir et al. (2022) [8] in their systematic review. The authors identify scalability as a multifaceted concept, encompassing not only network expansion but also enhancements in processing capabilities, memory, storage, and consensus strategies. Their work underscores the complexity of achieving scalability while maintaining the core benefits of blockchain technology.

In response to these challenges, several innovative approaches have been proposed. Lin et al. (2020) [9] introduced Rapido, a multi-path off-chain payment mechanism designed to address the overload and privacy issues inherent in single-path payment systems like the Lightning Network. By distributing payments across multiple paths, Rapido not only resolves the overload issue but also mitigates the skewness of payment channels, demonstrating a significant improvement in success rates compared to traditional approaches.

The concept of sharding has gained traction as a promising solution for blockchain scalability. Li et al. (2023) [10] provide a comprehensive survey of state-of-the-art sharding blockchains, analyzing various models, components, and potential attack surfaces. Their work highlights the potential of sharding to enhance scalability while maintaining security and decentralization, a crucial balance in blockchain design.

Addressing the scalability-security trade-off, Oliveira et al. (2020) [11] proposed the Blockchain Reputation-Based Consensus (BRBC) mechanism. This approach introduces a reputation score system for nodes, allowing only those with scores above a certain threshold to participate in block insertion. The authors demonstrate BRBC's resistance to various known attacks and its ability to expel malicious nodes efficiently, offering a novel perspective on consensus mechanisms in blockchain networks.

In the realm of practical implementations, Meeuw et al. (2020) [12] provide valuable insights from a real-world blockchain-managed microgrid in Switzerland. Their study empirically evaluates the feasibility of a Byzantine fault-tolerant blockchain system in a practical setting, highlighting the impact of communication infrastructure limitations on system performance. Their findings underscore the importance of considering hardware and network constraints in blockchain design, particularly for applications in resource-constrained environments.

The scalability of blockchain systems in distributed environments has also been explored. Gawande et al. (2022) [13] present groundbreaking work on scaling community detection algorithms on distributed-memory heterogeneous systems, including multi-GPU setups. While not directly focused on blockchain, their approach to parallelizing graph algorithms offers valuable insights for improving the efficiency of blockchain operations in distributed environments.

Otte et al. (2020) [14] introduced TrustChain, a Sybil-resistant scalable blockchain that offers an alternative to traditional proof-of-work mechanisms. By creating an immutable chain of temporally ordered interactions for each agent, TrustChain demonstrates how historical transaction records can provide security and scalability without requiring global consensus, a concept that could be adapted to enhance blockchain scalability.

The application of blockchain technology in specific domains, such as energy markets, has also been explored. Li and Zhang (2022) [15] developed a data-oriented distributed optimization strategy for large-scale HVAC systems, demonstrating the potential of distributed approaches in complex systems. While not directly related to blockchain, their work provides insights into distributed optimization techniques that could be relevant to blockchain scalability solutions.

Rawhouser et al. (2022) [16] examine the scaling of blockchain technology applications in developing countries, highlighting the importance of network effects and innovative scaling methods. Their analysis of approaches such as promoting technology platforms, leveraging collective action, and navigating institutional contexts offers valuable perspectives on the broader challenges of scaling blockchain solutions in diverse environments.

In conclusion, the literature reveals a multifaceted approach to addressing blockchain scalability, encompassing innovations in consensus mechanisms, network architecture, and application-specific optimizations. While significant progress has been made, challenges remain in balancing scalability with security, decentralization, and practical implementation constraints. Our work aims to build upon these foundations, specifically addressing the challenges of block finality verification in high-throughput blockchain systems like the NEAR Protocol.

3. Background

The development of our zero-knowledge proof system for block finality in the NEAR Protocol builds upon a rich foundation of cryptographic research and blockchain technology. This section provides an overview of the key concepts and related work that form the basis of our research.

3.1. Zero-knowledge proofs

Zero-knowledge proofs (ZKPs), first introduced by Goldwasser, Micali, and Rackoff in 1985 [17], are cryptographic protocols that allow one party (the prover) to prove to another party (the verifier) that a statement is true without revealing any information beyond the validity of the statement itself. ZKPs possess three fundamental properties [18, 19]:

- Completeness: If the statement is true, an honest verifier will be convinced by an honest prover.
- Soundness: If the statement is false, no cheating prover can convince an honest verifier that it is true, except with negligible probability.
- Zero-knowledge: If the statement is true, the verifier learns nothing other than the fact that the statement is true.

Recent advancements in ZKP systems, particularly in the development of succinct non-interactive zero-knowledge proofs (SNARKs) [20] and scalable transparent arguments of knowledge (STARKs) [21], have made ZKPs increasingly practical for real-world applications, including blockchain systems.

3.2. NEAR protocol

NEAR Protocol is a sharded, proof-of-stake blockchain designed to address the scalability limitations of earlier blockchain systems [4]. Key features of NEAR include:

- Nightshade sharding: A unique sharding approach that allows for horizontal scaling of the network's processing capacity.
- Doomslug consensus: A block production mechanism that enables rapid block finality.
- WebAssembly-based smart contracts: Allowing for more efficient and flexible smart contract execution.

Understanding NEAR's architecture is crucial for appreciating the challenges and opportunities in implementing a ZKP-based finality system.

3.3. Blockchain finality and light clients

Blockchain finality refers to the point at which a transaction or block can be considered irreversible. In probabilistic finality systems like Bitcoin, finality is achieved after a certain number of confirmations. In contrast, systems with deterministic finality, like NEAR, aim to provide quicker and more definitive transaction finality.

Light clients are crucial for broadening blockchain accessibility, allowing resource-constrained devices to interact with the blockchain without maintaining a full copy of the chain. However, traditional light client protocols often involve trade-offs between efficiency and security guarantees.

3.4. Related work in ZKP-based blockchain systems

Several projects have explored the application of ZKPs in blockchain systems:

- Zcash [22, 23]: Pioneered the use of zk-SNARKs for privacy-preserving transactions in a public blockchain.
- Coda Protocol (now Mina) [24, 25]: Implemented a succinct blockchain using the recursive composition of SNARKs.
- StarkWare [26, 27]: Developed STARKs for scalable, transparent, and post-quantum secure computation.

Our work builds upon these foundations, specifically addressing the challenges of block finality verification in the context of NEAR Protocol's high-throughput, sharded architecture.

3.5. Plonky2 framework

Our implementation leverages the Plonky2 framework, a state-of-the-art system for generating and verifying zero-knowledge proofs. Plonky2 combines [28–30]:

- The PLONK proof system: A universal and updateable structured reference string (SRS) scheme.
- FRI commitments: Providing a balance between proof size and verification time.
- Optimized arithmetization: Enhancing the efficiency of proof generation and verification.

Understanding Plonky2's capabilities and limitations is essential for contextualizing the performance characteristics of our system.

3.6. Cryptographic primitives

Our system relies on several fundamental cryptographic primitives:

- SHA-256: A widely-used cryptographic hash function, crucial for block hash verification.
- EdDSA (Edwards-curve Digital Signature Algorithm): An efficient digital signature scheme used for validator signature verification.

- Merkle trees: Fundamental data structures for efficient proof of membership, used in various components of our system.

The selection and implementation of these primitives significantly influence the security and performance of our ZKP system.

By building upon this diverse foundation of cryptographic research and blockchain technology, our work aims to address the critical challenges of scalability and security in modern blockchain systems, with a specific focus on enhancing light client capabilities in the NEAR Protocol ecosystem.

4. System architecture

The proposed zero-knowledge proof (ZKP) system for block finality in the NEAR Protocol represents a significant advancement in blockchain security and scalability. This section provides a comprehensive overview of the system's architecture, detailing its key components and their integration within the existing NEAR infrastructure. By leveraging the power of zero-knowledge proofs, our system enhances the security and efficiency of block verification while maintaining the decentralized nature of the blockchain.

4.1. High-level description of the ZKP-based block finality-proof system

At its core, our system generates succinct, non-interactive zero-knowledge proofs (SNARKs) that attest to the validity and finality of blocks in the NEAR blockchain. The system operates on a tri-block principle, utilizing data from three consecutive blocks to generate and verify proofs. This approach ensures a comprehensive validation of block data, signatures, and state transitions.

The proof generation process can be represented by the following function:

$$\Pi = \text{GenerateProof}(B_i, B_{i+1}, B_e, pk),$$

where: B_i is the current block being proven; B_{i+1} is the subsequent block; B_e is the previous epoch block; pk is the proving key; Π is the resulting zero-knowledge proof.

The verification process is then represented as:

$$0,1 = \text{VerifyProof}(\Pi, vk),$$

where vk is the verification key, and the output is a boolean indicating the validity of the proof.

This high-level structure allows for efficient verification of block finality without requiring verifiers to process the entire blockchain history.

4.2. Key components

Our system comprises four primary components, each responsible for verifying a critical aspect of block integrity and consensus:

4.2.1. Block hash verification

This component ensures the integrity of the block data by verifying the correctness of the block hash. It implements the SHA-256 hash function within the ZKP circuit, allowing for efficient proof generation and verification. The process can be represented as:

$$\pi_{\text{hash}} = \text{ProveHash}(B_i, H(B_i)),$$

where $H()$ is the SHA-256 hash function, and π_{hash} is the resulting proof.

4.2.2. Signature verification

This component validates the signatures of block producers, ensuring that the block has been properly approved by authorized validators. It implements the EdDSA (Edwards-curve Digital Signature Algorithm) over Curve25519. The verification process can be expressed as:

$$\pi_{\text{sig}} = \text{ProveSignature}(m, \sigma, pk),$$

where m is the message (typically the block hash), σ is the signature, pk is the public key, and π_{sig} is the resulting proof.

4.2.3. Validator key and stake verification

This component verifies that the block signers collectively represent at least two-thirds of the total stake in the network, a crucial aspect of NEAR's consensus mechanism. The process involves two main steps:

1. Proving the existence of validator keys in the validators list:
$$\pi_{\text{keys}} = \text{ProveValidKeys}(K_v, K_a),$$
where K_v is the set of all validator keys, and K_a is the set of actual signers.
2. Proving the sufficiency of stakes:
$$\pi_{\text{stakes}} = \text{ProveStakes}(S_v, S_a, T),$$
where S_v is the total stake, S_a is the stake of actual signers, and T is the two-thirds threshold.

4.2.4. Next block producer hash verification

This component ensures the integrity of the validator set for the upcoming epoch by verifying the correctness of the next_bp_hash stored in the current block. The process can be represented as:

$$\pi_{next_{bp}} = \text{ProveNextBP}(H(V_{next}), \text{next}_{\text{bphash}})$$

where V_{next} is the validator set for the next epoch, and $\text{next}_{\text{bphash}}$ is the hash stored in the current block.

4.3. Integration with NEAR Protocol's existing infrastructure

Our ZKP system is designed to seamlessly integrate with NEAR's existing blockchain infrastructure, complementing rather than replacing current validation mechanisms. The integration occurs at several key points:

- Block Production: During block production, in addition to standard NEAR block fields, producers generate ZK proofs for the previous block. These proofs are included in the new block's header.
- Block Propagation: When blocks are propagated through the network, the accompanying ZK proofs are transmitted alongside block data, allowing for rapid verification by receiving nodes.

- Light Client Synchronization: Light clients can utilize these ZK proofs to efficiently verify the blockchain state without downloading and processing the entire chain history.
- Cross-Shard Communication: In NEAR's sharded architecture, ZK proofs facilitate secure and efficient cross-shard state verification, enhancing the overall throughput and security of inter-shard transactions.
- Validator Set Transitions: The next block producer hash verification component ensures smooth and secure transitions between validator sets across epoch boundaries.

By integrating these crucial points, our ZKP system enhances NEAR's security and scalability without disrupting its core functionality. This approach allows for gradual adoption and provides backward compatibility with existing NEAR nodes and clients.

5. Zero-knowledge proof construction

The efficacy of our block finality proof system for the NEAR Protocol hinges on the robust construction of zero-knowledge proofs. This section delves into the cryptographic foundations of our system, detailing the primitives employed, the design of verification circuits, and the techniques used for proof aggregation. Our approach leverages state-of-the-art cryptographic tools to achieve a balance of security, efficiency, and scalability.

5.1. Cryptographic primitives used

Our system employs a carefully selected set of cryptographic primitives, each chosen for its security properties and efficiency in the context of zero-knowledge proofs.

5.1.1. SHA-256

We utilize the SHA-256 hash function for block hash verification and in the construction of Merkle trees. SHA-256 is defined as:

$$H(m) = \text{SHA-256}(m)\,,$$

where m is the input message and $H(m)$ is the resulting 256-bit hash.

The security of SHA-256 relies on its collision resistance and preimage resistance properties, making it suitable for our blockchain application.

5.1.2. EdDSA (Edwards-curve Digital Signature Algorithm)

For signature verification, we implement EdDSA over the Curve25519 elliptic curve. The EdDSA scheme consists of three main algorithms:

1. Key Generation: $(sk, pk) \leftarrow \text{KeyGen}(seed)$
2. Signing: $\sigma \leftarrow \text{Sign}(sk, m)$
3. Verification: $0,1 \leftarrow \text{Verify}(pk, m, \sigma)$,

where sk is the secret key, pk is the public key, m is the message, and σ is the signature.

EdDSA offers several advantages, including deterministic signatures, fast single-signature verification, and small public keys and signatures.

5.1.3. Plonky2 framework

Our implementation is built upon the Plonky2 framework, which combines the PLONK-proof system with optimized arithmetization and a custom commitment scheme. Plonky2 provides:

1. A universal and updateable structured reference string (SRS).
2. Efficient proof generation and verification.
3. Support for recursive proof composition.

The core of Plonky2 is based on the polynomial interactive oracle proof (IOP) model, which can be represented as:

$$(\pi, x) \leftarrow \text{Prove}(C, w)\,,\ 0,1 \leftarrow \text{Verify}(C, x, \pi)\,,$$

where C is the circuit, w is the witness, x is the public input, and π is the proof.

5.2. Circuit design for each verification component

Each component of our system requires a carefully designed arithmetic circuit to enable efficient zero-knowledge proof generation:

5.2.1. Block hash verification circuit

The block hash verification circuit implements the SHA-256 algorithm. It consists of:

- Message padding and chunking.
- Initialize hash values and round constants.
- Main compression function (64 rounds).
- Final addition.

The circuit is optimized to minimize the number of constraints while maintaining the security properties of SHA-256.

5.2.2. Signature verification circuit

Our EdDSA verification circuit comprises:

- Point decompression for the public key.
- Scalar multiplication for the signature.
- Point addition and equality check.

The circuit leverages the properties of the Edwards curve to optimize computations, particularly in the scalar multiplication step.

5.2.3. Validator key and stake verification circuit

This circuit consists of two main components:

- Merkle tree verification for proving key membership in the validator set.
- Arithmetic circuit for stake summation and threshold comparison.

The circuit is designed to efficiently handle varying numbers of validators while maintaining a fixed circuit size.

5.2.4. Next block producer hash verification circuit

Similar to the block hash verification circuit, this component implements SHA-256 but is specifically optimized for the fixed-size input of the validator list hash.

5.3. Proof aggregation techniques

To enhance the efficiency of our system, we employ several proof aggregation techniques:

5.3.1. Recursive SNARK Composition

We utilize recursive SNARK composition to aggregate proofs from different components. This technique allows us to verify the correctness of multiple sub-proofs within a single, succinct proof. The recursive composition can be represented as:

$$\pi_{\text{agg}} = \text{Aggregate}(\pi_1, \pi_2, \ldots, \pi_n),$$

where $\pi_1, \pi_2, \ldots, \pi_n$ are individual proofs and π_{agg} is the aggregated proof.

5.3.2. Batch verification

For signature verification, we implement a batch verification technique that allows multiple signatures to be verified simultaneously. This approach significantly reduces the overall computation required for verifying a large number of signatures.

5.3.3. Incremental aggregation

Our system employs an incremental aggregation approach, where proofs are combined as they are generated, rather than all at once. This technique is particularly beneficial for handling the dynamic nature of blockchain data.

The incremental aggregation process can be described as:

$$\pi_i = \text{Aggregate}(\pi_{i-1}, \pi_{\text{new}}),$$

where π_{i-1} is the previous aggregate proof, π_{new} is a newly generated proof, and π_i is the updated aggregate proof.

By leveraging these advanced cryptographic primitives and proof aggregation techniques, our system achieves a high degree of efficiency and scalability while maintaining strong security guarantees. The carefully designed circuits and aggregation methods allow for rapid proof generation and verification, crucial for the real-time demands of the NEAR Protocol's high-throughput blockchain.

6. Performance evaluation

Our zero-knowledge proof system for block finality in the NEAR Protocol aims to enhance the efficiency and security of light clients. This section presents a detailed analysis of our system's performance based on real-world testing data.

6.1. Experimental setup and methodology

Our experiments were conducted on a dedicated server with the following specifications:

- CPU: AMD Ryzen 9 7950X 16-Core Processor.
- Clock Speed: 4.7 GHz (base frequency: 4.5 GHz, max boost: 5.7 GHz).
- RAM: 64 GB DDR4.
- Storage: 1 TB NVMe SSD.
- Operating System: Ubuntu 20.04 LTS.

We implemented our system using the Rust programming language, leveraging the Plonky2 framework for zero-knowledge proof generation and verification. Our evaluation focused on four consecutive blocks from the NEAR blockchain, chosen to represent typical network activity.

6.2. Results and analysis

Our experimental results provide comprehensive insights into the performance characteristics of our zero-knowledge proof system for block finality in the NEAR Protocol. We analyze each component individually to understand its contribution to the overall system performance.

6.2.1. Block hash verification performance

Fig. 1 illustrates the performance metrics for block hash verification across four different blocks.

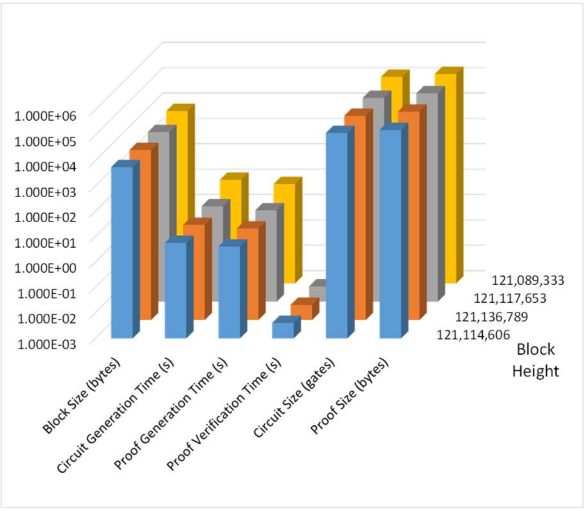

Figure 1: Block hash verification performance

The data reveals several important trends:

- Scalability: The circuit generation time and proof generation time exhibit a near-linear relationship with block size. This scalability is crucial for handling the varying block sizes typical in blockchain networks.

- Verification efficiency: Remarkably, the proof verification time remains consistently low (approximately 4 milliseconds) across all block sizes. This consistency is particularly advantageous for light clients, as it ensures rapid verification regardless of block complexity.
- Circuit complexity: The number of gates in the circuit grows with block size, reflecting the increased computational requirements for larger blocks. However, this growth is sublinear, indicating efficient circuit design.
- Proof size stability: The proof size remains constant at 165,684 bytes for most blocks, with only a slight increase to 180,112 bytes for the largest block. This stability is beneficial for network bandwidth considerations and storage requirements in light clients.

These results demonstrate that our block hash verification component achieves a balance between scalability and efficiency, particularly in the critical aspect of proof verification time.

6.2.2. Signature verification performance

Fig. 2 presents the performance data for signature verification across four block pairs.

The analysis of this data yields several significant observations:

- Consistency in proof generation: Despite variations in block size and, presumably, the number of signatures, the proof generation time remains remarkably consistent, ranging from 13.11 to 13.41 seconds. This consistency suggests that our system efficiently handles varying numbers of signatures without significant performance degradation.
- Rapid verification: The proof verification times are consistently low, ranging from 4.6 to 5.1 milliseconds. This speed is crucial for light clients, enabling rapid confirmation of signature validity.
- Proof size uniformity: The aggregated proof size remains constant at 133,080 bytes across all tested blocks. This uniformity is advantageous for predictable network bandwidth usage and storage requirements in light clients.
- Circuit generation variability: The circuit generation time shows some variation (11.55 to 14.40 seconds), likely due to differences in the complexity of signature data across blocks. However, this one-time cost does not impact the efficiency of subsequent proof verifications.

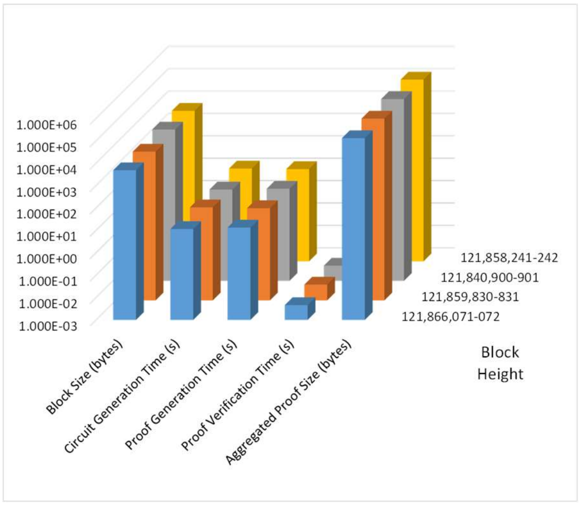

Figure 2: Signature verification performance

The performance characteristics of our signature verification component indicate its suitability for high-throughput blockchain systems, particularly in scenarios requiring frequent and rapid signature validations by light clients.

6.2.3. Validator key and stake verification performance

The validator key and stake verification component demonstrates efficient performance across various operations:

- Stake computation efficiency: The process of computing all stakes and checking the two-thirds threshold collectively accounts for approximately 0.0533 seconds, demonstrating the efficiency of our stake verification mechanism.
- Circuit building overhead: The time to build the circuit (0.0341 seconds) is comparable to the proof generation time (0.0588 seconds), indicating a well-balanced implementation.
- Overall efficiency: The total time for validator key and stake verification is approximately 0.1691 seconds, which is relatively low considering the complexity of the operations involved.

Table 1
Validator key and stake verification performance (for block 122,556,588-589)

Operation	Time (s)
Fulfilling validator values into the circuit	0.0006
Compute all stakes	0.0307
Compute 3 * valid_stake_sum	0.0003
Compute 2 * all_stake_sum	0.0226
Check if valid_stake_sum_3 >= all_stake_sum	0.0220
Circuit Build	0.0341
Proof Generation	0.0588

These results suggest that our system can efficiently handle the critical task of validator set verification, an essential component for maintaining the security of the consensus mechanism in proof-of-stake blockchains.

6.2.4. Next block producer hash verification performance

Fig. 3 displays the performance metrics for the next block producer hash verification.

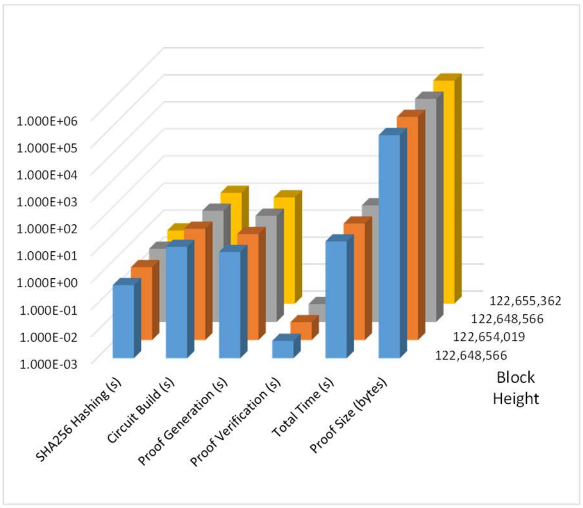

Figure 3: Next block producer hash verification performance

The analysis of this data reveals:

- Consistency: The total processing time shows remarkable consistency across different blocks, ranging from 20.5532 to 21.1767 seconds. This consistency is crucial for predictable performance in the NEAR Protocol.
- Efficient verification: Despite the relatively long proof generation times, the verification times are exceptionally low, consistently around 4.5 milliseconds. This efficiency is particularly beneficial for light clients, allowing for rapid verification of the next block producer set.
- Proof size stability: The proof size remains constant at 180,112 bytes across all tested blocks, which is advantageous for network bandwidth considerations and storage requirements.
- Component breakdown: The circuit building phase consistently accounts for the largest portion of the total time (approximately 60–63%), followed by proof generation (40–41%), with SHA256 hashing and proof verification taking significantly less time.

These results demonstrate that our next block producer hash verification component provides a robust and efficient mechanism for ensuring the integrity of validator set transitions, a critical aspect of maintaining long-term blockchain security.

6.2.5. Proof aggregation and final block verification performance

Fig. 4 presents the performance data for the final proof aggregation and block verification process.

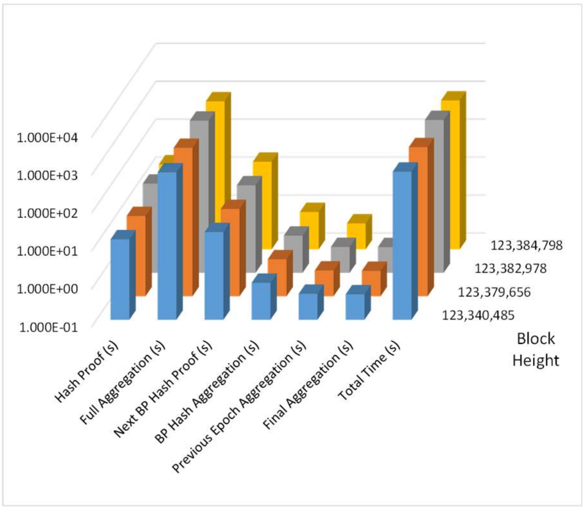

Figure 4: Final block verification performance

The analysis of this data yields several important insights:

- Dominance of full aggregation: The full aggregation process, which includes proving signatures, aggregating them, generating recursive proofs, and proving valid keys and stakes, consistently accounts for the vast majority of the total time (95–96%). This indicates that optimizing this step could significantly improve overall system performance.
- Consistency in other components: The time required for hash proof, next BP hash proof, and various aggregation steps remain relatively consistent across different blocks, contributing to predictable system behavior.
- Rapid final aggregation: The final aggregation step, crucial for light client verification, consistently takes less than 0.5 seconds. This efficiency is key to enabling quick block finality confirmation for light clients.
- Total processing time: The total proof generation time ranges from about 806 to 1073 seconds (approximately 13 to 18 minutes). While this may seem substantial, it's important to note that this process is performed by full nodes and does not affect the verification speed for light clients.
- Scalability considerations: The variation in total processing time across different blocks suggests that the system's performance scales with block complexity. However, the consistent final aggregation time indicates that this scaling does not significantly impact light client performance.

These results demonstrate that our proof aggregation and final block verification component effectively consolidates the various proofs into a single, efficiently verifiable proof, thereby enabling rapid block finality confirmation for light clients while maintaining the security guarantees of the underlying zero-knowledge proof system.

6.3. Comparison with existing solutions

When comparing our system to the native NEAR block verification process, we focus on the benefits for light clients:

- Verification time: Our system allows light clients to verify blocks in about 0.47 seconds, regardless of block complexity. This is a significant improvement over traditional light client verification, which may require multiple network round-trips and processing of block headers.
- Data transfer: Our system requires transferring only a constant-size proof (180,112 bytes) to light clients, regardless of block size or transaction count. This is substantially less than transferring full block data or even compressed block headers.
- Security guarantees: Our ZKP system provides cryptographic assurance of block validity, offering stronger security guarantees compared to traditional light client verification methods that may rely on assumptions about the network's honest majority.
- Computational requirements: While our proof generation is computationally intensive (13–18 minutes per block), this is performed by full nodes. Light clients only need to perform the much faster verification step (<0.5 seconds), making our system viable even for resource-constrained devices.

In conclusion, our ZKP-based system significantly reduces the computational and bandwidth requirements for light clients in the NEAR ecosystem, while enhancing security guarantees. The trade-off of increased block production time is outweighed by the benefits in light client efficiency and the potential for broader blockchain accessibility.

7. Discussion

The implementation and evaluation of our zero-knowledge proof system for block finality in the NEAR Protocol offer significant insights into the future of blockchain security and scalability. This section discusses the broader implications of our work, explores potential applications in other cybersecurity domains, and addresses limitations while outlining future research directions.

7.1. Implications for blockchain security and scalability

Our ZKP-based system demonstrates a promising approach to enhancing both the security and scalability of blockchain networks, particularly for light clients. The key implications include:

- Enhanced light client security: By providing succinct, verifiable proof of block validity, our system significantly reduces the trust assumptions required for light clients. This enhancement could lead to more secure and reliable mobile and IoT applications in the blockchain ecosystem.
- Improved scalability: The constant-size proofs and rapid verification times enable light clients to efficiently validate the blockchain state without downloading and processing the entire chain. This capability could dramatically increase the number of participants in blockchain networks without compromising decentralization.
- Cross-chain interoperability: The succinct nature of our ZKP system could facilitate more efficient and secure cross-chain communication, potentially accelerating the development of interoperable blockchain ecosystems.
- Reduced network overhead: By minimizing the data transfer required for block verification, our system could contribute to reduced network congestion and improved overall network efficiency in blockchain systems.

7.2. Potential applications in other cybersecurity domains

The principles and techniques developed in our ZKP system have potential applications beyond blockchain technology:

- Secure multiparty computation: Our approach to efficient proof aggregation could be adapted to enhance the performance and privacy of secure multiparty computation protocols in various cybersecurity applications.
- Privacy-preserving authentication: The zero-knowledge properties of our system could be leveraged to develop more robust and privacy-preserving authentication mechanisms for sensitive cybersecurity systems.
- Verifiable computing: The techniques used in our block verification system could be extended to create efficient verification mechanisms for outsourced computation in cloud computing environments.
- Secure software updates: Our ZKP system could be adapted to provide verifiable proof of the integrity and authenticity of software updates, enhancing security in software distribution systems.

7.3. Limitations and future work

While our system demonstrates promising results, several limitations and areas for future work remain:

1. Proof generation time: The substantial time required for proof generation (13–18 minutes per block) could pose challenges for real-time block production. Future work should focus on optimizing this process, possibly through parallel computation or more efficient cryptographic constructions.
2. Hardware requirements: The current implementation requires significant computational resources for proof generation. Research into hardware acceleration techniques could make the system more accessible to a broader range of network participants.

3. Post-quantum security: While our system provides strong security guarantees under current cryptographic assumptions, it is not yet post-quantum secure. Investigating post-quantum ZKP schemes is a crucial area for future research.
4. Dynamic sharding support: As NEAR Protocol moves towards dynamic sharding, our system will need to be adapted to efficiently handle cross-shard transactions and state transitions.
5. Formal verification: Developing formal proofs of the security properties of our ZKP system would provide stronger guarantees and increase confidence in its deployment in critical blockchain infrastructure.

8. Conclusions

This paper presents a novel zero-knowledge proof system for block finality in the NEAR Protocol, making several key contributions to the field of blockchain security and scalability:

- We developed a comprehensive ZKP-based verification system that encompasses block hash, signature, validator key and stake, and next block producer hash verification.
- Our system achieves constant-time verification for light clients, regardless of block size or complexity, significantly enhancing the efficiency and security of light client operations.
- We demonstrated the feasibility of using advanced cryptographic techniques, including the Plonky2 framework, to create practical ZKP systems for high-throughput blockchain networks.
- Our performance evaluation provides valuable insights into the scalability and efficiency of ZKP systems in real-world blockchain environments.

The development and successful implementation of our ZKP system for the NEAR Protocol point to a promising future for ZKP-based security in blockchain systems:

- We anticipate increased adoption of ZKP techniques in mainstream blockchain protocols, driven by the growing need for scalability and privacy in decentralized systems.
- Future research is likely to focus on reducing proof generation times and hardware requirements, making ZKP systems more accessible and practical for a wider range of blockchain applications.
- The integration of post-quantum cryptographic techniques with ZKP systems will become increasingly important as quantum computing advances threaten traditional cryptographic assumptions.
- Cross-chain interoperability solutions based on ZKP systems are poised to play a crucial role in the development of a more interconnected and efficient blockchain ecosystem.

In conclusion, our work demonstrates the potential of zero-knowledge proofs to address critical challenges in blockchain security and scalability. As the field continues to evolve, ZKP-based systems are set to become an integral component of next-generation blockchain architectures, enabling more secure, scalable, and privacy-preserving decentralized systems.

References

[1] H. Singh, Chapter 1. Decentralized Web, Distributed Ledgers, and Build-up to Blockchain, Distributed Computing to Blockchain, Academic Press (2023) 3–17. doi: 10.1016/B978-0-323-96146-2.00004-8.

[2] NEAR, Blockchains, Abstracted, (n. d.). URL: https://near.org/

[3] V. Zhebka, et al., Methodology for Choosing a Consensus Algorithm for Blockchain Technology, in: Digital Economy Concepts and Technologies, vol. 3665 (2024) 106–113.

[4] Sharding Design: Nightshade, NEAR Protocol (2023). URL: https://near.org/papers/nightshade/

[5] K. Kuznetsova, et al., Solving Blockchain Scalability Problem Using ZK-SNARK, Advances in Artificial Systems for Logistics Engineering III, Springer Nature Switzerland, Cham (2023) 360–371. doi: 10.1007/978-3-031-36115-9_33.

[6] O. Kuznetsov, et al., Implementing Recursive Proofs for Efficient Blockchain Verification: A zk-SNARKs Approach, Mathematical Modeling and Simulation of Systems, Springer Nature Switzerland, Cham (2024) 266–280. doi: 10.1007/978-3-031-67348-1_20.

[7] O. Kuznetsov, et al., Enhanced Security and Efficiency in Blockchain with Aggregated Zero-Knowledge Proof Mechanisms, IEEE Access 12 (2024) 49228–49248. doi: 10.1109/ACCESS.2024.3384705.

[8] M. H. Nasir, et al., Scalable Blockchains—A Systematic Review, Future Generation Comput. Syst. 126 (2022) 136–162. doi: 10.1016/j.future.2021.07.035.

[9] C. Lin, et al., Rapido: Scaling Blockchain with Multi-path Payment Channels, Neurocomputing 406 (2020) 322–332. doi: 10.1016/j.neucom.2019.09.114.

[10] Y. Li, J. Wang, H. Zhang, A survey of state-of-the-art sharding blockchains: Models, components, and attack surfaces, J. Netw. Comput. Appl. 217 (2023) 103686. doi: 10.1016/j.jnca.2023.103686.

[11] M. T. de Oliveira, et al., Blockchain Reputation-based Consensus: A Scalable and Resilient Mechanism for Distributed Mistrusting Applications, Comput. Netw. 179 (2020) 107367. doi: 10.1016/j.comnet.2020.107367.

[12] A. Meeuw, et al., Implementing a Blockchain-based Local Energy Market: Insights on Communication and Scalability, Comput. Commun. 160 (2020) 158–171. doi: 10.1016/j.comcom.2020.04.038.

[13] N. Gawande, et al., Towards Scaling Community Detection on Distributed-Memory Heterogeneous Systems, Parallel Computing 111 (2022) 102898. doi: 10.1016/j.parco.2022.102898.

[14] P. Otte, M. de Vos, J. Pouwelse, TrustChain: A Sybil-Resistant Scalable Blockchain, Future Generation Computer Systems 107 (2020) 770–780. doi: 10.1016/j.future.2017.08.048.

[15] Z. Li, J. Zhang, Data-oriented Distributed Overall Optimization for Large-Scale HVAC Systems with

Dynamic Supply Capability and Distributed Demand Response, Building and Environment 221 (2022) 109322. doi: 10.1016/j.buildenv.2022.109322.

[16] H. Rawhouser, et al., Scaling, Blockchain Technology, and Entrepreneurial Opportunities in Developing Countries, Journal of Business Venturing Insights 18 (2022) e00325. doi: 10.1016/j.jbvi.2022.e00325.

[17] S. Goldwasser, S. Micali, C. Rackoff, The Knowledge Complexity of Interactive Proof-Systems, in: 17th Annual ACM Symposium on Theory of Computing, Association for Computing Machinery (1985) 291–304. doi: 10.1145/22145.22178.

[18] U. Feige, A. Fiat, A. Shamir, Zero-Knowledge Proofs of Identity, J. Cryptology 1 (1988) 77–94. doi: 10.1007/BF02351717.

[19] A. Fiat, A. Shamir, How to Prove Yourself: Practical Solutions to Identification and Signature Problems, Advances in Cryptology (CRYPTO'86), Springer-Verlag (1987) 186–194.

[20] E. Ben-Sasson, et al., SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge, Advances in Cryptology (CRYPTO) (2013) 90–108. doi: 10.1007/978-3-642-40084-1_6.

[21] N. Bitansky, et al., Recursive Composition and Bootstrapping for SNARKS and Proof-Carrying Data, in: 45th Annual ACM Symposium on Theory of Computing, Association for Computing Machinery (2013) 111–120. doi: 10.1145/2488608.2488623.

[22] D. Hopwood, et al., Zcash Protocol Specification, Version 2022.3.8 [NU5] (2022).

[23] Privacy-Protecting Digital Currency, Zcash (n. d.). URL: https://z.cash/

[24] N. Ariffin, A. Z. Ismail, The Design and Implementation of Trade Finance Application based on Hyperledger Fabric Permissioned Blockchain Platform, International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) (2019) 488–493. doi: 10.1109/ISRITI48646.2019.9034576.

[25] Y. N. Patil, et al., A Decentralized and Autonomous Model to Administer University Examinations, Blockchain Technology for IoT Applications, Springer (2021) 119–134. doi: 10.1007/978-981-33-4122-7_6.

[26] StarkWare, STARK Math: The Journey Begins, StarkWare (2020). URL: https://medium.com/starkware/stark-math-the-journey-begins-51bd2b063c71

[27] StarkWare, Recursive STARKs, StarkWare (2022). URL: https://medium.com/starkware/recursive-starks-78f8dd401025

[28] Introducing Plonky2 (n. d.). URL: https://polygon.technology/blog/introducing-plonky2

[29] mir-protocol/plonky2 (n. d.). URL: https://github.com/mir-protocol/plonky2/tree/main

[30] Plonky2: Fast Recursive Arguments with PLONK and FRI (2023). URL: https://github.com/mir-protocol/plonky2/blob/main/plonky2/plonky2.pdf