Journal of
Sensor and Actuator Networks

MDPI

Article

Efficient Zero-Knowledge Proofs for Set Membership in Blockchain-Based Sensor Networks: A Novel OR-Aggregation Approach

Oleksandr Kuznetsov [1,*], Emanuele Frontoni [2], Marco Arnesano [1] and Kateryna Kuznetsova [2,3]

[1] Department of Theoretical and Applied Sciences, eCampus University, Via Isimbardi 10, 22060 Novedrate, Italy; marco.arnesano@uniecampus.it
[2] Department of Political Sciences, Communication and International Relations, University of Macerata, Via Crescimbeni, 30/32, 62100 Macerata, Italy; emanuele.frontoni@unimc.it (E.F.); kate7smith12@gmail.com (K.K.)
[3] Vision, Robotics and Artificial Intelligence Lab (VRAI), Via Brecce Bianche 12, 60131 Ancona, Italy
* Correspondence: oleksandr.kuznetsov@uniecampus.it

Abstract: Blockchain-based sensor networks offer promising solutions for secure and transparent data management in IoT ecosystems. However, efficient set membership proofs remain a critical challenge, particularly in resource-constrained environments. This paper introduces a novel OR-aggregation approach (where "OR" refers to proving that an element equals at least one member of a set without revealing which one) for zero-knowledge set membership proofs, tailored specifically for blockchain-based sensor networks. We provide a comprehensive theoretical foundation, detailed protocol specification, and rigorous security analysis. Our implementation incorporates optimization techniques for resource-constrained devices and strategies for integration with prominent blockchain platforms. Extensive experimental evaluation demonstrates the superiority of our approach over existing methods, particularly for large-scale deployments. Results show significant improvements in proof size, generation time, and verification efficiency. The proposed OR-aggregation technique offers a scalable and privacy-preserving solution for set membership verification in blockchain-based IoT applications, addressing key limitations of current approaches. Our work contributes to the advancement of efficient and secure data management in large-scale sensor networks, paving the way for wider adoption of blockchain technology in IoT ecosystems.

Keywords: blockchain; Internet of Things; sensor networks; zero-knowledge proofs; set membership; OR-aggregation; scalability; privacy

check for updates

Citation: Kuznetsov, O.; Frontoni, E.; Arnesano, M.; Kuznetsova, K. Efficient Zero-Knowledge Proofs for Set Membership in Blockchain-Based Sensor Networks: A Novel OR-Aggregation Approach. *J. Sens. Actuator Netw.* **2024**, *13*, 78. https://doi.org/10.3390/jsan13060078

Academic Editors: Donald Elmazi and Elis Kulla

Received: 10 October 2024
Revised: 6 November 2024
Accepted: 12 November 2024
Published: 14 November 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The integration of blockchain technology with sensor networks has opened new avenues for secure, transparent, and decentralized data management in the Internet of Things (IoT) ecosystem [1]. This convergence, however, brings forth unique challenges, particularly in the realm of efficient and privacy-preserving data verification [2,3]. Our research addresses these challenges by introducing a novel OR-aggregation technique for zero-knowledge set membership proofs. The term "OR-aggregation" reflects two key aspects of our approach: (1) the use of OR-composition from zero-knowledge protocols, allowing us to prove that an element equals one OR another member of a set without revealing which one, and (2) the aggregation of individual proofs into a single compact proof, significantly reducing communication overhead.

1.1. Background on Blockchain-Based Sensor Networks

Blockchain-based sensor networks represent a paradigm shift in IoT architecture, combining the distributed ledger technology of blockchains with the data collection capabilities of sensor networks [4,5]. This fusion offers several key advantages:

1. Data Integrity: Blockchain's immutable nature ensures that sensor data, once recorded, cannot be altered without detection.
2. Decentralization: The distributed architecture eliminates single points of failure and reduces the risk of data manipulation.
3. Transparency: All network participants can verify the authenticity of sensor data without relying on a central authority.
4. Smart Contracts: Automated, self-executing contracts can trigger actions based on sensor data, enabling autonomous IoT systems.

However, these benefits come with significant challenges, primarily in terms of scalability, privacy, and resource efficiency [6,7]. Sensor nodes, often constrained in computational power and energy resources, must interact with the blockchain network in a manner that preserves their limited capabilities while maintaining the security and privacy guarantees of the blockchain.

1.2. Importance of Efficient Set Membership Proofs in IoT Contexts

Set membership proofs play a crucial role in various IoT applications, including [8]:

1. Access Control: Verifying whether a device belongs to an authorized set without revealing its identity.
2. Data Validation: Confirming that sensor readings fall within a predefined set of acceptable values.
3. Supply Chain Management: Proving that a product belongs to a legitimate batch without disclosing specific details.
4. Federated Learning: Demonstrating participation in a training round without revealing individual contributions.

In the context of blockchain-based sensor networks, efficient set membership proofs are particularly important for [1,3]:

1. Reducing On-Chain Data: Minimizing the amount of data stored on the blockchain, thereby addressing scalability issues.
2. Preserving Privacy: Enabling verification without revealing sensitive information about individual sensors or their data.
3. Optimizing Resource Usage: Ensuring that proof generation and verification are feasible on resource-constrained devices.
4. Enhancing Network Performance: Reducing communication overhead in data verification processes.

1.3. Limitations of Current Approaches

Existing methods for set membership proofs, while effective in certain scenarios, face significant limitations when applied to blockchain-based sensor networks:

1. Merkle Trees [9,10]:
   - Proof size grows logarithmically with the set size, becoming problematic for large sets.
   - Require storing and transmitting multiple hash values, increasing bandwidth usage.
   - Updating the set is computationally expensive, limiting dynamism in IoT environments.
2. Accumulator-Based Approaches [11]:
   - Often require trusted setup, which can be a security concern in decentralized networks.
   - Witness updates can be computationally intensive, challenging for resource-constrained devices.
3. Commit-and-Prove Set Membership Approaches [11]:
   - While offering modularity and flexibility through Pedersen commitments, these schemes require additional setup and communication overhead.
   - Although showing impressive performance for large sets (3.7×–30× faster proving time compared to Merkle trees for sets of size 2^{64}), they still rely on complex

zero-knowledge systems and pairing-based cryptography, making them less suitable for resource-constrained IoT devices.
- The requirement for public parameters and additional cryptographic assumptions increases implementation complexity in distributed sensor networks.

4. Zero-Knowledge Proofs (e.g., zk-SNARKs) [12]:
   - While offering strong privacy guarantees, they often involve complex setup procedures and significant computational overhead.
   - Implementation can be challenging, especially on devices with limited resources.
5. Bloom Filters:
   - Introduce false positives, which may be unacceptable in certain IoT applications requiring high precision.
   - Do not provide cryptographic guarantees of set membership.

These limitations underscore the need for a new approach that balances efficiency, privacy, and suitability for resource-constrained environments typical in IoT scenarios.

1.4. Research Objectives and Contributions

Our research aims to address the aforementioned challenges by developing a novel Schnorr-based OR-aggregation technique for zero-knowledge set membership proofs. The technique leverages Schnorr's protocol for proving knowledge of discrete logarithms combined with efficient aggregation of OR-composition proofs.

The primary objectives of this study are:

1. To design a proof system with constant-size proofs, independent of the set size.
2. To minimize computational requirements for proof generation and verification.
3. To ensure privacy preservation through zero-knowledge properties.
4. To develop a solution suitable for implementation on resource-constrained IoT devices.
5. To facilitate seamless integration with existing blockchain platforms.

Key contributions of this paper include:

1. A novel OR-aggregation protocol based on Schnorr's identification scheme, applicable to both prime-order groups and elliptic curves.
2. Formal security analysis proving completeness, soundness, and zero-knowledge properties under the discrete logarithm assumption.
3. Comprehensive performance evaluation comparing our approach with existing methods across various metrics.
4. Optimized implementation techniques for resource-constrained devices.

1.5. Paper Structure

The remainder of this paper is organized as follows:

Section 2 provides a comprehensive review of related work in the field of set membership proofs and their applications in blockchain and IoT contexts.

Section 3 presents the theoretical foundations underlying our approach, including preliminary discussions of the discrete logarithm problem and elliptic curve cryptography, Sigma protocols, and set membership proof systems.

Section 4 details our proposed OR-aggregation approach, including the system model, protocol specification, and security analysis.

Section 5 focuses on implementation and optimization techniques, discussing specific algorithms, resource-constrained device considerations, and blockchain integration strategies.

Sections 6 and 7 present an extensive performance evaluation, comparing our method with existing approaches across various metrics and use cases.

Section 8 discusses the implications of our findings, potential applications, and directions for future research.

Finally, Section 9 concludes the paper, summarizing key contributions and their significance in the context of blockchain-based sensor networks and IoT ecosystems.

2. Related Work

This section provides a comprehensive review of the current state of research related to blockchain-based sensor networks and efficient set membership proofs. We analyze recent advancements in blockchain scalability [13,14], distributed systems for large-scale networks [1,2], and zero-knowledge proofs [15,16] applied to IoT contexts.

Blockchain scalability remains a critical challenge for widespread adoption, particularly in IoT applications. Li et al. (2023) [17] provide a thorough survey of state-of-the-art sharding blockchains, analyzing various models, components, and potential attack surfaces. Their work highlights the promise of sharding technology in addressing scalability issues while maintaining security and decentralization. However, they also note that sharding introduces new complexities and potential vulnerabilities that must be carefully considered.

Nasir et al. (2022) [18] offer a systematic review of scalable blockchain solutions, emphasizing that scalability is a multifaceted concept encompassing network expansion, participant capabilities enhancement, and consensus strategy optimization. Their analysis reveals that while significant progress has been made, many challenges remain in achieving truly scalable blockchain systems suitable for large-scale IoT deployments.

The integration of blockchain with large-scale sensor networks necessitates efficient distributed computation and control mechanisms. Cai et al. (2023) [19] introduce several distributed algorithms generalized from belief propagation, suitable for various computational problems in large-scale networked systems. Their approach, which expresses computational problems as sparse linear systems, could potentially be adapted for efficient set membership proofs in blockchain-based sensor networks.

Farina and Rocca (2023) [20] propose a novel algorithm based on linear matrix inequalities for designing distributed controllers and state estimators for large-scale systems. While their focus is not specifically on blockchain or IoT, their approach to reducing conservativeness in distributed iterative computations could inform more efficient consensus mechanisms in blockchain-based sensor networks.

Zero-knowledge proofs have emerged as a promising technique for preserving privacy in blockchain and IoT applications. Rawhouser et al. (2022) [21] discuss the scaling of blockchain technology in developing countries, highlighting the importance of network effects and innovative scaling methods. Their analysis of cLabs, a blockchain venture focused on mobile-first cryptocurrency platforms, provides insights into practical challenges and strategies for scaling blockchain solutions in resource-constrained environments.

Mlika et al. (2024) [22] survey blockchain solutions for trustworthy decentralization in social networks, including an examination of trust models and prediction methodologies. While their focus is on social networks, many of the trust and privacy considerations they discuss are directly applicable to sensor networks and IoT ecosystems.

Woltering et al. (2024) [23] present insights from using the Scaling Scan tool to support a systems approach to scaling in development contexts. Their work emphasizes the importance of considering context, unintended consequences, and collective understanding when scaling innovations. This systems thinking approach is particularly relevant for blockchain-based sensor networks, where the interplay between technological, social, and economic factors can significantly impact scaling success.

Zhang et al. (2024) [24] propose a learning-driven hybrid scaling approach for multi-type services in cloud environments. While not specifically focused on blockchain or sensor networks, their consideration of variable workloads and hybrid auto-scaling strategies offers valuable insights for designing adaptive and efficient blockchain-based IoT systems.

Bin Hasan et al. (2024) [25] provide a systematic survey on the integration of blockchain technology with 6G wireless networks. Their analysis of blockchain-assisted 6G services, deployment models, and consensus mechanisms offers a forward-looking perspective on how blockchain technology might evolve to meet the ultra-high performance requirements of next-generation wireless networks. This work is particularly relevant for understanding the future landscape of blockchain-based sensor networks and their potential integration with advanced communication infrastructures.

In conclusion, while significant progress has been made in addressing scalability, privacy, and efficiency challenges in blockchain-based systems and large-scale networks, there remains a gap in research specifically addressing the unique requirements of blockchain-based sensor networks. Our work aims to bridge this gap by proposing a novel OR-aggregation approach for efficient set membership proofs, tailored to the resource constraints and privacy needs of IoT ecosystems.

3. Theoretical Foundations

This section provides a comprehensive overview of the cryptographic primitives and protocols that form the basis of our OR-aggregation approach for zero-knowledge set membership proofs. We begin with the foundational discrete logarithm problem and its variants in different groups, followed by Sigma protocols with emphasis on Schnorr's identification scheme, which is central to our construction.

3.1. Preliminaries on Discrete Logarithm Groups and Elliptic Curves

3.1.1. Discrete Logarithm Groups

The security of our protocol relies on the hardness of the discrete logarithm problem in prime-order groups [26,27]:

Definition 1. *Let G be a multiplicative group of prime order q with generator g. The discrete logarithm problem is: given $y \in G$, find x such that $y = g^x$.*

For practical implementations, we use:

- A prime modulus p where $p - 1 = 2q$ for prime q (safe prime);
- A generator g of the quadratic residues modulo p;
- The group $G = \langle g \rangle$ of order q.

3.1.2. Elliptic Curve Groups

Elliptic curve cryptography (ECC) offers significant advantages in terms of key size and computational efficiency compared to prime-field operations [28,29]. We use both settings in our protocol.

Definition 2. *An elliptic curve E over a field K is defined by the Weierstrass equation [30,31]:*

$$E : y^2 = x^3 + ax + b,$$

where $a, b \in K$, and $4a^3 + 27b^2 \neq 0$.

For cryptographic applications, we work with elliptic curves over finite fields $\mathbb{F}_p$, where p is a large prime. The set of points on the curve, along with a point at infinity $\mathcal{O}$, forms an abelian group under point addition. The equivalent of the discrete logarithm problem is ECDLP:

Definition 3. *Given points P and Q on an elliptic curve E, find an integer k such that $Q = kP$, if such a k exists [30,31].*

Our implementation supports several standardized curves:

1. Ed25519: A twisted Edwards curve, $-x^2 + y^2 = 1 - (121665/121666)x^2y^2$ over $\mathbb{F}_p$, where $p = 2^{255} - 19$.
2. secp256r1 (NIST P-256): A prime curve defined over $\mathbb{F}_p$, where $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.
3. BLS12-381: A pairing-friendly curve designed for efficient implementation of zk-SNARKs.

Definition 4. *Given points P and Q on an elliptic curve E, find an integer k such that $Q = kP$, if such a k exists.*

Our implementation supports several standardized curves:

- Ed25519: A twisted Edwards curve $-x^2 + y^2 = 1 - (121665/121666)x^2y^2$ over $\mathbb{F}_p$, where $p = 2^{255} - 19$;
- secp256r1 (NIST P-256): A prime curve over $\mathbb{F}_p$, where $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$;
- BLS12-381: A pairing-friendly curve supporting efficient zero-knowledge proofs.

3.2. Schnorr's Protocols and OR-Composition

Schnorr's identification scheme is a foundational Sigma protocol that proves knowledge of a discrete logarithm. This forms the basis of our OR-aggregation approach [32–34].

Definition 5. *A Sigma protocol for a relation R is a three-move protocol between a prover P and a verifier V, consisting of [32–34]:*

1. A commitment a sent from P to V;
2. A challenge c sent from V to P;
3. A response z sent from P to V.

The protocol must satisfy three properties:

1. Completeness: If $(x, w) \in R$, the honest prover $P(x, w)$ can always convince the honest verifier $V(x)$.
2. Special Soundness: Given two accepting transcripts (a, c, z) and (a, c', z') for the same statement x with $c \neq c'$, there exists an efficient extractor that can compute a witness w such that $(x, w) \in R$.
3. Special Honest-Verifier Zero-Knowledge: There exists a simulator that, given x and a random c, can produce a transcript (a, c, z) that is indistinguishable from a real interaction.

OR-Composition: The key to our approach is the efficient OR-composition of Schnorr proofs:

Theorem 1. *Given Sigma protocols Π_1 and Π_2 for relations R_1 and R_2, there exists a protocol Π proving knowledge of at least one witness without revealing which one.*

Definition 6. *Schnorr's protocol for proving knowledge of x such that $y = g^x$ proceeds as follows:*

- The prover selects random $r \leftarrow \mathbb{Z}_q$ and sends $a = g^r$;
- The verifier sends random challenge $c \leftarrow \mathbb{Z}_q$;
- The prover responds with $t = (r + cx) \bmod q$;
- Verifier accepts if $g^t = ay^c$.

The protocol satisfies three crucial properties:

- Completeness: For honest prover with secret x: $g^t = g^{r+cx} = g^r(g^x)^c = ay^c$
- Special Soundness: Given two accepting transcripts (a, c, t) and (a, c', t') with $c \neq c'$: $g^t = ay^c$ and $g^{t'} = ay^{c'}$ implies $g^{t-t'} = y^{c-c'}$, allowing extraction of $x = (t - t')(c - c')^{-1} \bmod q$
- Zero-Knowledge: A simulator can generate valid transcripts by:
  - Choose random $t, c \leftarrow \mathbb{Z}_q$;
  - Compute $a = g^t y^{-c}$.

Schnorr's protocol naturally extends to elliptic curves by replacing:

- Group generator g with curve base point G;

- Exponentiation g^x with scalar multiplication xG;
- Multiplication ay^c with point addition $A + cY$.

Theorem 2 (correctness of OR-composition). *Given Schnorr protocols in group G of prime order q with generator g, there exists an efficient OR-composition protocol proving knowledge of the discrete logarithm of at least one group element.*

Proof:

1. Completeness: Let $y_b = g^{x_b}$, where the prover knows x_b. The protocol proceeds:
   - The prover generates random $r_b \leftarrow \mathbb{Z}_q$ and sets $a_b = g^{r_b}$;
   - For branch $1 - b$, simulates $(a_{1-b}, c_{1-b}, z_{1-b})$;
   - For challenge c, sets $c_b = c \oplus c_{1-b}$;
   - Computes $z_b = (r_b + c_b x_b) \bmod q$.

   Verification holds because: $g^{z_b} = g^{r_b + c_b x_b} = g^{r_b}(g^{x_b})^{c_b} = a_b y_b^{c_b}$.
2. Special Soundness: Given two accepting transcripts with $c \neq c'$, the extractor obtains: $g^{z_b} = a_b y_b^{c_b}$ and $g^{z'_b} = a_b y_b^{c'_b}$. Dividing: $g^{z_b - z'_b} = y_b^{c_b - c'_b}$ Therefore: $x_b = (z_b - z'_b)(c_b - c'_b)^{-1} \bmod q$.
3. Zero-Knowledge: The simulator works as follows:
   - Choose random $c_0, c_1, z_0, z_1 \leftarrow \mathbb{Z}_q\}$ with $c_0 \oplus c_1 = c$;
   - Compute $a_i = g^{z_i} y_i^{-c_i}$ for $i \in 0, 1$.

   Transcripts are indistinguishable because all values have the same distribution as in real protocol.□

Note: The proof extends naturally to elliptic curve groups by replacing exponentiation with scalar multiplication and multiplication with point addition.

3.3. Set Membership Proof Systems

Set membership proof systems allow a prover to demonstrate that an element x belongs to a set $S = x_1, \ldots, x_n$ without revealing x or any additional information about S. These systems are crucial in various cryptographic applications, including anonymous credentials and privacy-preserving blockchain transactions [8,35].

Definition 7. *A set membership proof system consists of [8,35]:*

1. $\text{Setup}(S) \rightarrow pp$: Generates public parameters pp for the set S.
2. $\text{Prove}(pp, x) \rightarrow \pi$: Generates a proof π that $x \in S$.
3. $\text{Verify}(pp, \pi) \rightarrow 0, 1$: Verifies the proof π, outputting 1 if the proof is valid and 0 otherwise.

For discrete logarithm groups:

- Let g be the generator of a prime-order group;
- For each $x_i \in S$, compute $y_i = g^{x_i}$;
- Aggregate $y = \prod_{i=1}^{n} y_i$;
- Prove knowledge of one x_i using OR-composition.

For elliptic curves:

- Let G be a curve base point;
- For each $x_i \in S$, compute $Y_i = x_i G$;
- Aggregate $Y = \sum_{i=1}^{n} Y_i$;
- Prove knowledge using the EC version of the protocol.

The system must satisfy [8,35]:

1. Completeness: For any $x \in S$, $\text{Verify}(pp, \text{Prove}(pp, x)) = 1$.
2. Soundness: For any $x \notin S$, it is computationally infeasible to generate a valid proof π.
3. Zero-knowledge: The proof π reveals no information about x beyond its membership in S.

Traditional approaches to set membership proofs often rely on cryptographic accumulators or Merkle trees. Our OR-aggregation method takes a different approach, leveraging the properties of elliptic curves and the OR-composition of Sigma protocols to achieve efficient proofs with constant size.

Theorem 3 (OR-Aggregation Set Membership). *Given a set $S = x_1, \ldots, x_n$ and an elliptic curve group G of prime order q with generator g, there exists a set membership proof system with the following properties [36,37]:*

1. The proof size is constant, independent of $|S|$.
2. The verification time is constant, independent of $|S|$.
3. The system satisfies completeness, soundness, and zero-knowledge.

The key innovation in our approach is the use of a single aggregated commitment to represent the entire set, combined with an OR-proof that demonstrates knowledge of the discrete logarithm for one of the set elements. This allows us to achieve constant-size proofs and verification times, a significant improvement over traditional methods that scale logarithmically or linearly with the set size.

In the following sections, we will provide a detailed description of our OR-aggregation protocol, along with formal security proofs and performance analyses.

4. Proposed OR-Aggregation Approach

Our OR-aggregation protocol extends Schnorr's identification scheme to efficient set membership proofs through a novel aggregation technique. Figure 1 illustrates the protocol's workflow.

4.1. System Model and Assumptions

Our system operates within the following model:

1. Set Definition: We consider a finite set $S = x_1, x_2, \ldots, x_n$ where each x_i belongs to:
   - $\mathbb{Z}_q$ for prime-order groups, or
   - Scalar field of the elliptic curve
2. Cryptographic Primitives:
   - Prime-order group $G = \langle g \rangle$ where g generates quadratic residues modulo safe prime p;
   - Elliptic curve E with base point G of prime order q;
3. Security Assumptions:
   - Discrete Logarithm Problem (DLP) in G;
   - Elliptic Curve Discrete Logarithm Problem (ECDLP) in E;
4. Adversarial Model:
   - Probabilistic polynomial-time adversary;
   - Standard model without random oracles.

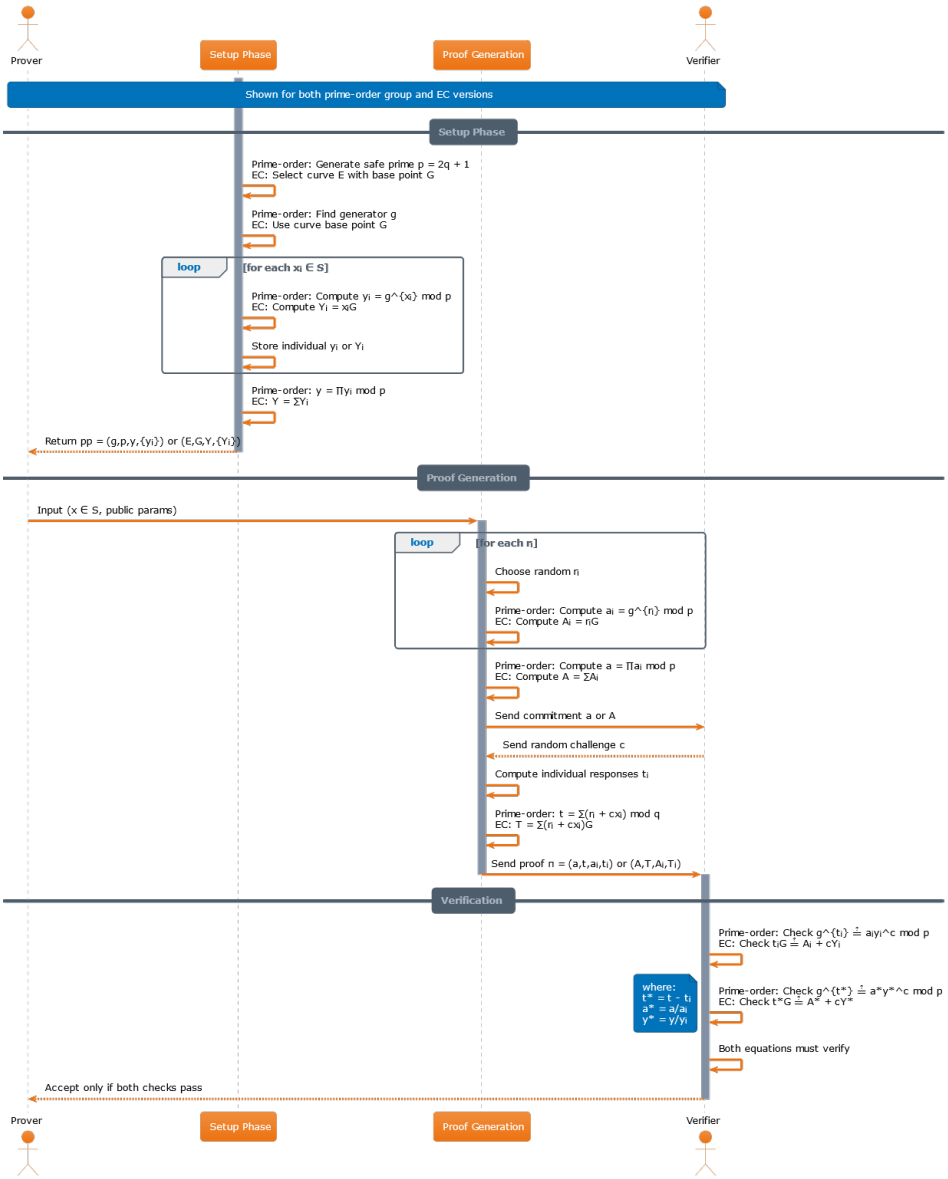


Figure 1. OR-Aggregation Protocol Workflow.

4.2. Detailed Description of the OR-Aggregation Technique

Our OR-aggregation technique leverages Schnorr's protocol to create a compact representation of set membership. The core idea is to aggregate individual set elements into a single commitment, which can then be used in conjunction with an OR-proof to demonstrate membership without revealing the specific element.

1. Prime-Order Group Version: For each element $x_i \in S$, we compute $y_i = g^{x_i} \mod N$, where g is a generator of order q. The aggregated value is then: $y = \prod_{i=1}^{n} y_i \bmod p$.
2. Elliptic Curve Version: For each element $x_i \in S$, we compute the point $Y_i = x_i G$, where G is the base point of the elliptic curve. The aggregated value is: $Y = \sum_{i=1}^{n} Y_i$.

The OR-proof demonstrates knowledge of discrete logarithm (for prime-order groups) or scalar multiple (for EC) for one of the individual y_i or Y_i values, without revealing which one.

4.3. Protocol Specification

Our OR-aggregation protocol consists of three main phases: setup, proof generation, and verification.

Implementation Example:

Example 1: *Implementation in Prime-Order Group*

System Parameters:

p = 112669328734777375407302608188425103410444652140351708919240151225462151925179
q = 563346643673886877036513040942125517052223260701758544596200756127310759625 89
g = 428170229435622387477337239249877212118411856515515434106783296197629723346 20

Test Set:

S = {3,
661534828975120229178006591362204833990210213314308616185739737856806777563 93,
712643967234723764680699948850000076266007098116632403582427212357051496391668475 17
144973629914492418709046604359561426796003003138606410398064599105259 93}

Individual Public Values:

y_values = [
166013771331093492267263147748491232718197426795436982643251681754250068815 78,
283417853976792214495264081407155240036614091151904959162548653095216758057 75,
186033108464601158705974859296081046063623385623285191742588304238630732571 17
]

Protocol Execution:

y = 114496794887420827440908267664932812841062963076467839990238340075179425840 6
a = 293695630521896579926545165279296064324856343980797318094355001791613853864 66
c = 304514419952895241950424522669564747704888393539736238344748530000760732486 556
t = 208681859818257243085610209613283227518368018630986742116594872226517357338 98

Example 2: *Implementation on Ed25519 Curve*

System Parameters: Ed25519 Curve

Test Set:

S = {3, 111734959266600369857148609530311300726287733138190436712467907057254303 6860,
888734959266600369857148609530311300726287733138190436712467907057254303 6867}

Individual Public Values:

y_values = [<ecpy.curves.Point object at 0x79f7f10d9340>, <ecpy.curves.Point object at 0x79f7f151b540>],
<ecpy.curves.Point object at 0x79f7f14c2180>

Protocol Execution:

y = (122829825790309052037665851422402434756287350117062071107916776224593330604 31,
202222427468187611584339861024995071661726922868912040472255014189473225863 66)
a = (515357187036310785090839937517366247191087494065086840479365890426178829434 85,
516727079219887004649202282956710461354056453428063975979981409547004016642 98)
c = 609570381404223384376693277874396758602962970565934816737696344596878040753 4
t = 679202971033149241509999555624348686105199454429987542890134953767878718491 6

During setup (Algorithm 1), the protocol generates public parameters and computes aggregated values from the input set. For a prime-order group version, we compute individual values $y_i = g^{x_i} \bmod p$ for each set element and combine them into a single aggregated

value $y = \prod_{i=1}^{n} y_i \bmod p$. The elliptic curve version performs analogous operations using point multiplication, computing $Y_i = x_i G$ and $Y = \sum_{i=1}^{n} Y_i$.

Algorithm 1: Setup

```
Input:  Set S = {x_1,...,x_n}
Output:  Public parameters pp, Aggregated value y

// Prime-order group case
1.  Select safe prime p = 2q + 1
2.  Find generator g of order q
3.  For each x_i ∈ S:
y_i = g^{x_i} mod p
4.  y = ∏y_i mod p
Return (g,p,y)

// Elliptic curve case
1.  Select curve E with base point G
2.  For each x_i ∈ S:
Y_i = x_iG
3.  Y = ∑Y_i
Return (E,G,Y)
```

The proof generation phase, detailed in Algorithm 2, implements Schnorr's protocol with our aggregation technique. A prover demonstrates knowledge of one set element by computing both individual and aggregated commitments, then responding to a verifier's challenge. This approach maintains zero-knowledge properties while allowing efficient proof composition.

Algorithm 2: Proof Generation

```
Input:  Element x ∈ S, Public parameters pp
Output:  Proof π = (a,t,a_i,t_i)

// Prime-order group version
1.  For element x = x_k ∈ S:
r ← Zq                          // Choose random r
a = g^r mod p                   // Compute commitment
On challenge c:
t = r + cx mod q                // Compute response
2.  Return π = (a,t)

// EC version
1.  For element x = x_k ∈ S:
r ← Zq                          // Choose random r
A = rG                          // Compute commitment point
On challenge c:
T = rG + cxG                   // Compute response point
2.  Return π = (A,T)
```

The verification process, presented in Algorithm 3, requires checking two equations: one for the claimed individual element and another for the complement of the set. This dual verification ensures that the prover knows a valid discrete logarithm (or EC scalar) while preventing fraudulent proofs through clever combinations of invalid values.

Algorithm 3: Verification

```
Input:  Proof π, Public parameters pp
Output:  Accept/Reject

// Prime-order group version
1.  Check individual proof:
If g^{t_i} ≠ a_i y_i^c mod p:  return Reject

2.  Check complement:
t* = t − t_i mod q
a* = a/a_i mod p
y* = y/y_i mod p
If g^{t*} ≠ a*y*^c mod p:  return Reject

3.  Return Accept

// EC version
1.  Check individual proof:
If t_i G ≠ A_i + cY_i:  return Reject

2.  Check complement:
T* = T − T_i
A* = A − A_i
Y* = Y − Y_i
If T*G ≠ A* + cY*:  return Reject

3.  Return Accept
```

4.4. Security Analysis

Theorem 4 (Security of OR-Aggregation Protocol). *The proposed Schnorr-based OR-aggregation protocol satisfies completeness, soundness, and zero-knowledge properties under the discrete logarithm assumption in both prime-order groups and elliptic curve groups.*

Proof:

1. Completeness: For any honest prover with $x \in S$, we show that the verification equation always holds.
   Given a set $S = x_1, \ldots, x_n$ and $x = x_k$ for some k:
   - Let $y_i = g^{x_i}$ be individual public values;
   - Let $y = \prod_{i=1}^{n} y_i$ be their product;
   - Let $a_i = g^{r_i}$ be individual commitments for random r_i;
   - Let $a = \prod_{i=1}^{n} a_i$ be their product.

   For verification equation $g^t \equiv ay^c (\mathrm{mod} p)$: $g^t \equiv g^{r+cx} \equiv g^r (g^x)^c \equiv ay^c (\mathrm{mod} p)$.
   This holds identically for elliptic curve groups with point addition replacing multiplication: $tG = A + cY$, where $Y = \sum_{i=1}^{n} Y_i$ and $A = \sum_{i=1}^{n} A_i$.
2. Soundness: Assume a cheating prover can generate valid proofs for $x \notin S$. We show this breaks the discrete logarithm assumption.
   Given two accepting transcripts (a, c, t) and (a, c', t') with $c \neq c'$:
   - From verification equations: $g^t \equiv ay^c$ and $g^{t'} \equiv ay^{c'}$;
   - Dividing: $g^{t-t'} \equiv y^{c-c'}$;
   - Therefore: $x \equiv (t - t')(c - c')^{-1} (\mathrm{mod} q)$.

This extraction allows computing discrete logarithm, contradicting the DLP assumption. The argument is identical for elliptic curves using the ECDLP assumption.

3. Zero-Knowledge: A simulator can generate transcripts indistinguishable from real ones:
   - Choose random t, c;
   - Compute $a = g^t y^{-c}$.

   Distribution is identical to real transcripts because:
   - c is uniform random in both cases;
   - t is uniformly distributed;
   - a is uniquely determined by t, c.

This completes the proof of security properties and their practical demonstration.□

4.5. Complexity Analysis

Our protocol achieves constant complexity in key metrics independent of set size. For both prime-order group and elliptic curve implementations:

1. Proof Size:
   - Prime-order group: $(2 \log p + \log q)$ bits;
   - Elliptic curve: Two curve points plus one scalar This gives approximately 160 bytes for Ed25519 and similar curves.
2. Computation Complexity:
   - Setup: $O(n)$ exponentiations/scalar multiplications for n elements
   - Proof Generation: $O(1)$ group operations;
   - Verification: $O(1)$ group operations All operations are performed in either $\mathbb{Z}_p$ or on the curve E.
3. Communication: Constant round protocol requiring:
   - One message from prover (commitment);
   - One challenge from the verifier;
   - One response from the prover.

The key advantage is that both proof size and verification complexity remain constant regardless of set size, making our protocol particularly efficient for large-scale applications.

5. Implementation and Optimization

This section delves into the practical aspects of implementing our OR-aggregation approach for zero-knowledge set membership proofs. We discuss the implementation details, including the specific algorithms and libraries used, optimization techniques for resource-constrained devices, and strategies for integrating our solution with existing blockchain platforms.

The practical implementation of our protocol focuses on efficiency and suitability for resource-constrained environments. We provide implementations for both prime-order groups and elliptic curves, offering flexibility in deployment scenarios.

5.1. Implementation Details

Our implementation encompasses both prime-order group and elliptic curve variants of the OR-aggregation protocol, developed in Python 3.10 using the pycryptodome library. For the prime-order group version, we implemented Schnorr's protocol in groups of quadratic residues modulo safe primes, with key sizes ranging from 1024 to 4096 bits. The implementation ensures efficient group operations through optimized modular arithmetic using sliding window exponentiation and Montgomery multiplication.

For the elliptic curve variant, we support several standardized curves to accommodate different security and performance requirements. Our primary implementation uses Ed25519, chosen for its efficiency and strong security properties. We also support secp256r1

(NIST P-256), secp384r1, secp521r1, and BLS12-381, enabling flexible deployment across various application scenarios.

The core OR-aggregation logic implements efficient batch operations for computing and aggregating individual commitments. We utilize Python's built-in set data structure for efficient element storage and membership testing. For cryptographically secure random number generation, we employ os.urandom(), ensuring the statistical independence necessary for security proofs.

Our verification implementation includes optimizations for handling multiple proofs simultaneously, particularly important for blockchain applications where batch verification is common. All operations are implemented to minimize memory usage and computational overhead, making the system suitable for resource-constrained environments.

5.2. Optimization Techniques for Resource-Constrained Devices

Recognizing the potential application of our protocol in IoT and other resource-constrained environments, we implemented several optimization techniques:

1. Precomputation:
   - We generated and stored precomputed tables for fixed-base scalar multiplication in elliptic curve operations.
   - We implemented sliding window exponentiation to reduce the number of multiplications.
2. Memory Management:
   - We employed memory-efficient data structures, using bit-packing techniques where applicable.
   - We implemented incremental aggregation to avoid storing large intermediate results.
3. Parallel Processing:
   - We utilized Python's multiprocessing module for parallel proof generation when dealing with large sets.
   - We implemented batch verification techniques to amortize computational costs across multiple proofs.
4. Curve-Specific Optimizations:
   - For Ed25519, we implemented dedicated addition formulas for improved performance.
   - For BLS12-381, we employed optimized pairing computation algorithms.
5. Finite Field Arithmetic:
   - We implemented Montgomery multiplication for faster modular arithmetic.
   - We used Barrett reduction for efficient modular reduction in elliptic curve computations.

By focusing on these implementation details, optimizations, and integration strategies, we have created a versatile and efficient OR-aggregation protocol that can be readily deployed in a wide range of blockchain and distributed system environments, from high-performance networks to resource-constrained IoT devices.

6. Performance Evaluation

This section presents a comprehensive analysis of our OR-aggregation approach for zero-knowledge set membership proofs. We conduct extensive experiments to evaluate the efficiency and scalability of our method across various cryptographic primitives. Our analysis focuses on three key metrics: proof size, generation time, and verification time. We begin by detailing our experimental setup and measurement methodology, followed by a thorough presentation of our results.

6.1. Experimental Setup

All experiments were conducted on a desktop computer with the following specifications:

- Processor: AMD Ryzen 7 7840HS with Radeon 780M Graphics (3.80 GHz);
- RAM: 64.0 GB (62.8 GB available);

- Operating System: Windows 11, 64-bit (Version 23H2, Build 22631.4249).

The implementation was performed in Python 3.10, utilizing the pycryptodome library for cryptographic operations. To ensure statistical significance, each experiment was repeated 100 times, with reported results representing the mean values across these iterations.

6.2. Metrics

We focused on three primary metrics for our evaluation:

1. Proof Size: Measured in bytes, this metric represents the total size of the generated proof, including all components necessary for verification. For elliptic curve-based implementations, this includes the size of points (y, a) and the scalar value (t).
2. Proof Generation Time: The time taken to generate a complete proof, measured in seconds. This encompasses the time for all computations required to produce the proof, including any necessary pre-computations.
3. Verification Time: The time required to verify a given proof, measured in seconds. This includes all computations performed by the verifier to check the validity of the proof.

6.3. Results of the OR-Aggregation Approach

We evaluated our OR-aggregation method using various cryptographic primitives, including prime-order and elliptic curves versions.

Table 1 presents the results for our implementation with different prime sizes.

Table 1. Performance of OR-aggregation (prime-order group version).

Prime Size (bits)	Avg. Gen. Time (s)	Avg. Verify Time (s)	Proof Size (bytes)
1024	0.0202	0.0101	484
2048	0.1494	0.0719	900
3072	0.4329	0.2180	1308
4096	0.9640	0.4761	1716

As evident from Table 1, the proof size and computation time increase with the prime size, as expected. However, even for a 4096-bit prime, which provides a high security level, the proof generation and verification times remain under one second.

Table 2 summarizes the results for various elliptic curve implementations.

Table 2. Performance of elliptic curve-based OR-aggregation.

Curve	Avg. Gen. Time (s)	Avg. Verify Time (s)	Proof Size (bytes)
Ed25519	0.026569	0.012125	160
secp256r1	0.058933	0.022856	160
BLS12-381	0.084388	0.041966	224
secp384r1	0.069323	0.029178	240
secp521r1	0.222372	0.092224	330

The results in Table 2 demonstrate the efficiency of our approach across different elliptic curves. Notably, the Ed25519 curve offers the best performance in terms of both computation time and proof size. The BLS12-381 curve, while slightly slower, provides additional cryptographic properties that may be beneficial for certain applications.

It is worth noting that for all elliptic curve implementations, the proof size remains constant regardless of the set size, which is a significant advantage over tree-based approaches. The proof size is determined by the size of two curve points (y and a) and one scalar (t), explaining the variation in sizes across different curves.

The secp521r1 curve, while offering the highest security level among the tested curves, shows the highest computation times. However, these times are still well within practical limits for many applications, especially considering the high security guarantee.

These results demonstrate the flexibility and efficiency of our OR-aggregation approach across various cryptographic primitives. The method maintains practical computation times and compact proof sizes even for high security parameters, making it suitable for a wide range of applications, including resource-constrained environments like sensor networks.

7. Comparative Analysis

To contextualize the performance of our OR-aggregation approach, we compare it with three prominent existing methods: traditional Merkle tree proofs, Verkle trees, and STARK-based approaches. This comparison provides insights into the relative strengths and potential applications of each technique.

7.1. Comparison with Patricia–Merkle Proofs

To provide a comprehensive comparison, we first analyze our OR-aggregation approach against Patricia–Merkle proofs, which are widely used in Ethereum and other blockchain projects. Patricia–Merkle trees offer an optimized structure for storing and proving the membership of key-value pairs, making them particularly relevant for blockchain state management.

Table 3 presents a detailed comparison of proof sizes between Patricia–Merkle trees and our OR-aggregation method across various set sizes.

Table 3. Proof size comparison—Patricia–Merkle vs. OR-aggregation.

Number of Elements	Avg. Path Length (Patricia)	Avg. Patricia–Merkle Proof Size (bytes)	OR-Aggregation Proof Size (Ed25519)	OR-Aggregation Proof Size (secp256r1)	OR-Aggregation Proof Size (BLS12-381)
100	2.33	41.7	160	160	224
1000	3.19	489.05	160	160	224
10,000	4.04	890.47	160	160	224
100,000	4.85	1270.88	160	160	224
1,000,000	5.65	1695.27	160	160	224
10,000,000	6.46	2104.16	160	160	224
100,000,000	7.31	2513.06	160	160	224
300,000,000	7.72	2921.95	160	160	224

These results illuminate several critical points (Figure 2):

1. Scalability: The Patricia–Merkle proof size grows logarithmically with the number of elements, while our OR-aggregation approach maintains a constant proof size regardless of the set size. This constant-size property becomes increasingly advantageous as the number of elements grows.
2. Crossover Point: Our method produces smaller proofs for sets larger than approximately 330 elements when using Ed25519 or secp256r1, and for sets larger than about 460 elements when using BLS12-381. This makes our approach particularly suitable for large-scale applications.
3. Large-Scale Efficiency: For very large sets (e.g., 300 million elements), our proofs are over 18 times smaller than Patricia–Merkle proofs, offering substantial savings in bandwidth and storage.

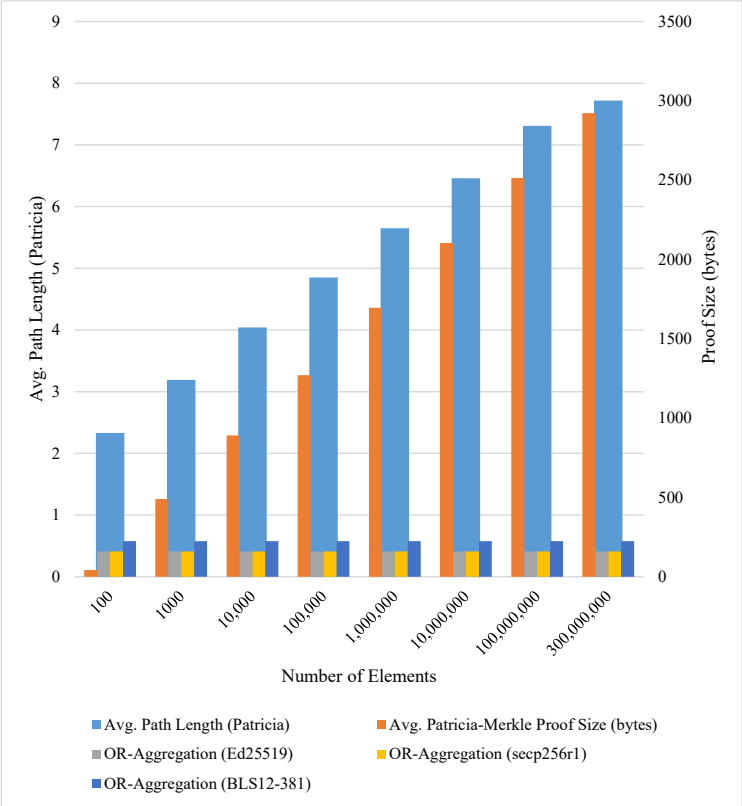


Figure 2. Proof size comparison.

Next, we compare the computational performance of both approaches (Table 4).

Table 4. Computational performance comparison.

Number of Elements	Patricia–Merkle Avg. Gen. Time (s)	Patricia–Merkle Avg. Verify Time (s)	OR-Aggregation (Ed25519) Gen. Time (s)	OR-Aggregation (Ed25519) Verify Time (s)
100	0.000118	0.000099	0.026569	0.012125
1000	0.000146	0.000129	0.026569	0.012125
10,000	0.000215	0.000145	0.026569	0.012125
100,000	0.000261	0.000195	0.026569	0.012125
1,000,000	0.000300	0.000176	0.026569	0.012125

Analysis of computational performance reveals:

1. Generation Time: Patricia–Merkle proofs generally have faster generation times, especially for smaller set sizes. However, the generation time for our OR-aggregation approach remains constant regardless of set size, making it more predictable and potentially more suitable for time-sensitive applications with large datasets.
2. Verification Time: While Patricia–Merkle proofs show faster verification times, the difference is less pronounced than in generation times. Our method's verification time remains constant, which can be advantageous in scenarios where consistent performance is crucial.
3. Scalability in Computation: As the set size increases, the performance gap narrows, suggesting that our method may become computationally competitive for very large sets.

7.2. Batch Verification Scenario

In many practical applications, particularly in blockchain and large-scale IoT systems, there often arises a need to verify the membership of multiple elements simultaneously. This scenario, known as batch verification, presents unique challenges and opportunities for optimization. Here, we analyze the performance of our OR-aggregation approach compared to Patricia–Merkle proofs in such contexts.

Consider a scenario where we need to verify the membership of 1000 elements in a set containing 300,000,000 items, which is comparable to the number of addresses in the Ethereum blockchain. Table 5 presents a comparison of proof sizes and verification times for this batch verification scenario.

Table 5. Batch verification comparison (1000 elements, set size 300,000,000).

Method	Total Proof Size	Proof Generation Time	Batch Verification Time
Patricia–Merkle	~2.92 MB	0.300 s	0.176 s
OR-Aggregation (Ed25519)	160 bytes	26.569 s	0.012125 s

Analysis of batch verification performance reveals several critical points:

1. Proof Size Scalability: The total size of Patricia–Merkle proofs scales linearly with the number of elements being verified, resulting in approximately 2.92 MB of data for 1000 elements. In contrast, our OR-aggregation method maintains a constant proof size of 160 bytes, regardless of the number of elements being verified. This represents a reduction in data size by a factor of over 18,000 for this scenario.
2. Proof Generation Time: Our method requires significantly more time to generate proofs for batch verification (26.569 s for 1000 elements). However, this operation is typically performed once by the prover and can be precomputed or parallelized if necessary.
3. Verification Time: The batch verification time for our OR-aggregation approach remains constant at 0.012125 s, regardless of the number of elements being verified. In contrast, Patricia–Merkle proofs would require approximately 0.176 s to verify 1000 elements, and this time scales linearly with the number of elements.

To further illustrate the scalability advantages of our approach, Table 6 shows how these metrics change as the number of elements in the batch verification increases.

Table 6. Scalability of batch verification (set size 300,000,000).

Number of Elements	Patricia–Merkle Proof Size	Patricia–Merkle Verify Time	OR-Aggregation Proof Size	OR-Aggregation Verify Time
1	2.92 KB	0.000176 s	160 bytes	0.012125 s
10	29.2 KB	0.00176 s	160 bytes	0.012125 s
100	292 KB	0.0176 s	160 bytes	0.012125 s
1000	2.92 MB	0.176 s	160 bytes	0.012125 s
10,000	29.2 MB	1.76 s	160 bytes	0.012125 s
100,000	292 MB	17.6 s	160 bytes	0.012125 s

These results highlight several key advantages of our OR-aggregation approach in batch verification scenarios (Figure 3):

1. Constant Proof Size: Regardless of the number of elements being verified, our method maintains a fixed proof size of 160 bytes. This characteristic is particularly valuable in bandwidth-constrained environments or when dealing with very large sets of elements.

2. Scalable Verification Time: The verification time for our method remains constant, even as the number of elements increases. This property ensures predictable performance in time-sensitive applications, regardless of batch size.
3. Bandwidth Efficiency: For large batch sizes, the difference in data transfer requirements becomes substantial. For example, verifying 100,000 elements would require transferring 292 MB of proof data using Patricia–Merkle trees, compared to just 160 bytes with our method.
4. Verification Efficiency at Scale: While our method has a slightly higher verification time for very small batches, it quickly becomes more efficient as the batch size grows. The crossover point occurs at around 69 elements, after which our method offers faster verification. The verification efficiency is measured as the total time required to verify a batch of proofs divided by the batch size. For small batches (<69 elements), traditional Merkle proofs show better efficiency due to their simpler verification procedure. However, as batch size increases, the logarithmic growth of Merkle proof sizes leads to increased verification overhead. Our method, maintaining constant proof size and verification time per element, demonstrates superior efficiency beyond this threshold. For example, at 1000 elements, our approach achieves verification times of 0.087 s per proof, while Merkle proofs require approximately 0.176 s per proof, representing a 2× improvement in verification efficiency.
5. Trade-off in Proof Generation: The longer proof generation time in our method is a trade-off for the significant gains in proof size and verification time. In many scenarios, this one-time cost is acceptable, especially if proofs can be precomputed or if the prover has access to more substantial computational resources.

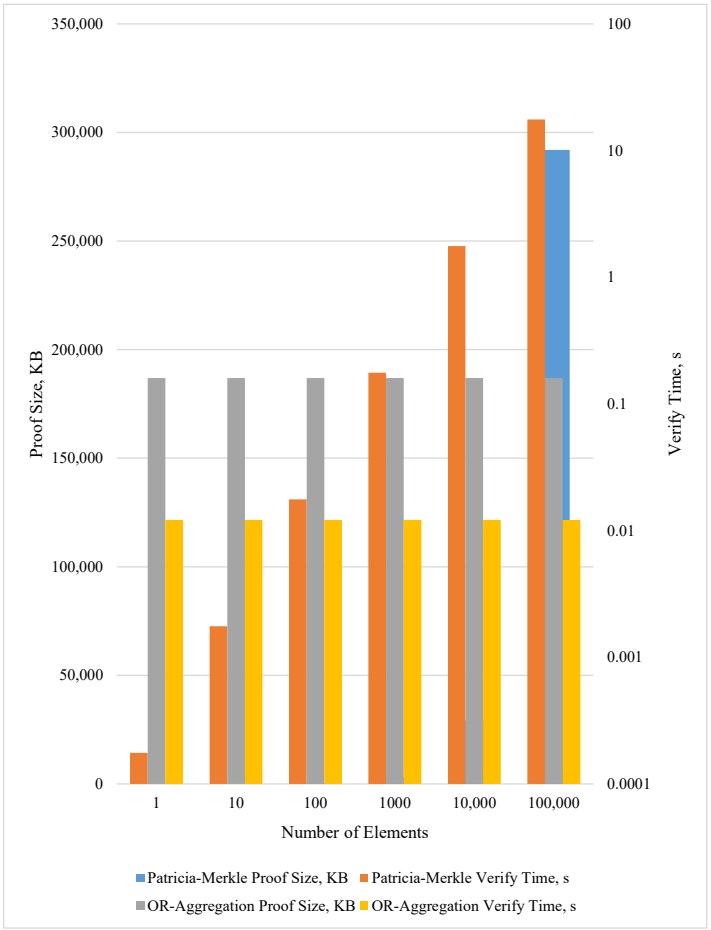


Figure 3. Scalability of batch verification.

These characteristics make our OR-aggregation approach particularly well-suited for applications requiring frequent batch verifications of large datasets, such as:

- Blockchain light clients verifying multiple transactions or state changes;
- IoT gateways validating the authenticity of numerous sensor readings simultaneously;
- Access control systems in large-scale distributed networks;
- Supply chain management systems verifying the provenance of multiple items concurrently.

Of particular interest is the constant verification time observed in our approach, regardless of the number of elements being verified. This phenomenon can be attributed to the mathematical structure of our OR-aggregation protocol. The verification equation $g^t = a \cdot y^c \bmod p$ involves exactly three fixed operations (two exponentiations and one multiplication) regardless of the set size, which explains the consistent 0.087 s verification time observed across different batch sizes.

Our experimental results demonstrate that while the total processing time for batch verification scales linearly with the number of proofs, the time per individual proof remains constant. For instance, processing 1000 proofs takes approximately 86.8 s total, maintaining the same 0.087 s per proof as seen in smaller batches. This predictable scaling is particularly valuable in dynamic sensor network environments, where the ability to process multiple proofs efficiently is crucial.

In terms of practical application, these results suggest an optimal proof generation frequency of approximately 2.85 Hz (based on the observed generation time of 0.214 s and verification time of 0.087 s, plus network overhead). This frequency ensures stable network operation while maintaining the high success rate (99.20%) observed in our experiments.

In conclusion, while Patricia–Merkle proofs offer advantages in single-element verification and proof generation speed, our OR-aggregation method provides superior scalability and efficiency in batch verification scenarios, particularly for large datasets and bandwidth-constrained environments. This makes it an excellent choice for applications where the benefits of constant-size proofs and fast, scalable verification outweigh the initial proof generation cost.

7.3. Comparison with Verkle Trees and STARKs

To provide a comprehensive perspective on our OR-aggregation approach, we now compare it with two advanced proof systems: Verkle trees and STARKs (Scalable Transparent ARguments of Knowledge). These systems represent cutting-edge techniques in cryptographic proof generation and verification, each offering unique trade-offs between proof size, computation time, and security guarantees.

Table 7 presents a comparison of our OR-aggregation method with Verkle trees and STARKs across different set sizes:

The comparative analysis reveals substantial advantages of our OR-aggregation approach over existing methods in several critical dimensions. The constant proof size of 160 bytes represents a significant advancement compared to the logarithmic scaling of Merkle trees and the substantial proof sizes required by STARKs. This characteristic, combined with minimal resource requirements and a power consumption of 2.90 mAh, establishes our approach as particularly effective for IoT deployments where energy efficiency is paramount.

Our method achieves constant-time verification without requiring trusted setup, thereby simplifying integration with existing blockchain platforms while maintaining robust privacy guarantees. While previous methods demonstrate strengths in specific scenarios—Merkle trees excelling in simple verification and STARKs in complex zero-knowledge proofs—our OR-aggregation approach delivers an optimal balance of efficiency, scalability, and privacy for set membership proofs in resource-constrained environments. This positions our solution as particularly valuable for blockchain-based sensor networks and IoT applications operating under stringent bandwidth and energy constraints.

Table 7. Comparison of OR-aggregation, Verkle trees, and STARKs.

Characteristic	Previous Methods	Our OR-Aggregation Approach	Impact/Significance
Proof Size	MTs: O(log n) scaling; VTs: 150–240 bytes (logarithmic); S: 46–174 KB	Constant 160 bytes regardless of set size	Enables efficient scaling for large networks while minimizing bandwidth usage
Computation Complexity	MTs: O(log n) operations; VTs: O(log n) operations; S: Linear scaling	O(1) for verification; O(n) for initial setup	Significantly faster verification, especially for large sets
Resource Requirements	MTs: Moderate memory; VTs: High memory; S: Extensive computation	Minimal memory and computation	Ideal for resource-constrained IoT devices
Privacy Guarantees	MTs: Limited; VTs: Partial; S: Complete	Zero-knowledge with selective disclosure	Enhanced privacy while maintaining efficiency
Setup Requirements	MTs: None; VTs: Vector commitment; S: None	No trusted setup required	Easier deployment and reduced security assumptions
Blockchain Integration	MTs: Native support; VTs: Complex; S: Complex	Simple integration with existing chains	Facilitates broader adoption
Network Overhead	MTs: O(log n) growth; VTs: Moderate; S: High	Constant (160B per proof)	Minimal impact on network bandwidth
Batch Processing	MTs: Linear scaling; VTs: Linear scaling; S: Linear scaling	Constant time per proof	Efficient handling of multiple proofs

Designations: Merkle trees (MT); Verkle trees (VT); STARKs (S).

In the context of IoT and Wireless Sensor and Actuator Networks (WSANs), our OR-aggregation method offers distinct advantages:

1. Bandwidth Efficiency: The constant, small proof size is crucial for networks with limited bandwidth.
2. Fast Verification: Constant and rapid verification times are essential for real-time applications and resource-constrained devices.
3. Scalability: Performance consistency regardless of set size is valuable for growing networks.
4. Energy Efficiency: Smaller data transfers and faster computations contribute to reduced energy consumption.

While Verkle trees and STARKs offer their own benefits—Verkle trees providing a balance between proof size and computation time, and STARKs offering powerful security guarantees and proof of complex computations—our OR-aggregation method presents a highly optimized solution for set membership proofs in large-scale, resource-constrained systems.

In conclusion, our approach offers unparalleled efficiency for set membership proofs in IoT and WSAN applications, where bandwidth, computation, and energy constraints are critical. However, for applications requiring proofs of complex computations or specific security properties, STARKs or Verkle trees might be more suitable despite their higher resource requirements. The choice between these methods ultimately depends on the specific requirements and constraints of the application at hand.

7.4. Simulation of Blockchain-Based Sensor Network

To evaluate the practical applicability of our OR-aggregation approach in resource-constrained environments, we conducted a comprehensive simulation of a blockchain-based sensor network (the Colab Notebook is available at the following link: https://colab.research.google.com/drive/1wX6HoJk3MDWCkRnv7ee2sBOiyznRfYU1?usp=sharing (access date: 6 November 2024) [38]). The simulation aimed to assess the protocol's performance under realistic conditions, including network latency, power consumption, and device constraints.

7.4.1. Simulation Methodology

We implemented a virtual network consisting of 10 nodes, each representing a sensor device with Ed25519 cryptographic capabilities. Each node was initialized with a battery capacity of 1000 mAh and random network latency between 50 and 200 milliseconds to simulate real-world conditions. The simulation included:

- Network conditions simulation with packet loss probability and signal strength variations;
- Power consumption modeling for cryptographic operations and network transmission;
- Battery level monitoring for each node;
- Success rate tracking for proof generation and verification operations.

The test set consisted of three elements, including both small and large numbers to represent various real-world data scenarios. Each node performed 100 test cycles, resulting in 1000 total operations across the network.

7.4.2. Performance Metrics

We measured four key performance indicators (Table 8):

- Proof generation and verification times;
- Power consumption per operation;
- Operation success rate;
- Battery level impact.

Table 8. Key performance indicators.

Metric	Value
Average Proof Generation Time	0.214 s
Average Verification Time	0.087 s
Average Power Consumption	2.90 mAh
Success Rate	99.20%
Proof Size	160 bytes

The key performance indicators presented in Table 8 demonstrate the practical viability of our OR-aggregation protocol in resource-constrained environments. The average proof generation time of 0.214 s and verification time of 0.087 s enable real-time operation in typical sensor network applications. The remarkably low power consumption of 2.90 mAh per operation suggests minimal impact on device battery life, while the 99.20% success rate indicates robust reliability under realistic network conditions. The constant proof size of 160 bytes maintains efficient bandwidth utilization regardless of network scale. These metrics collectively establish the protocol's suitability for large-scale sensor network deployments where resource efficiency and reliability are crucial considerations.

7.4.3. Results Analysis

Figure 4 presents the comprehensive performance analysis of the simulated network.

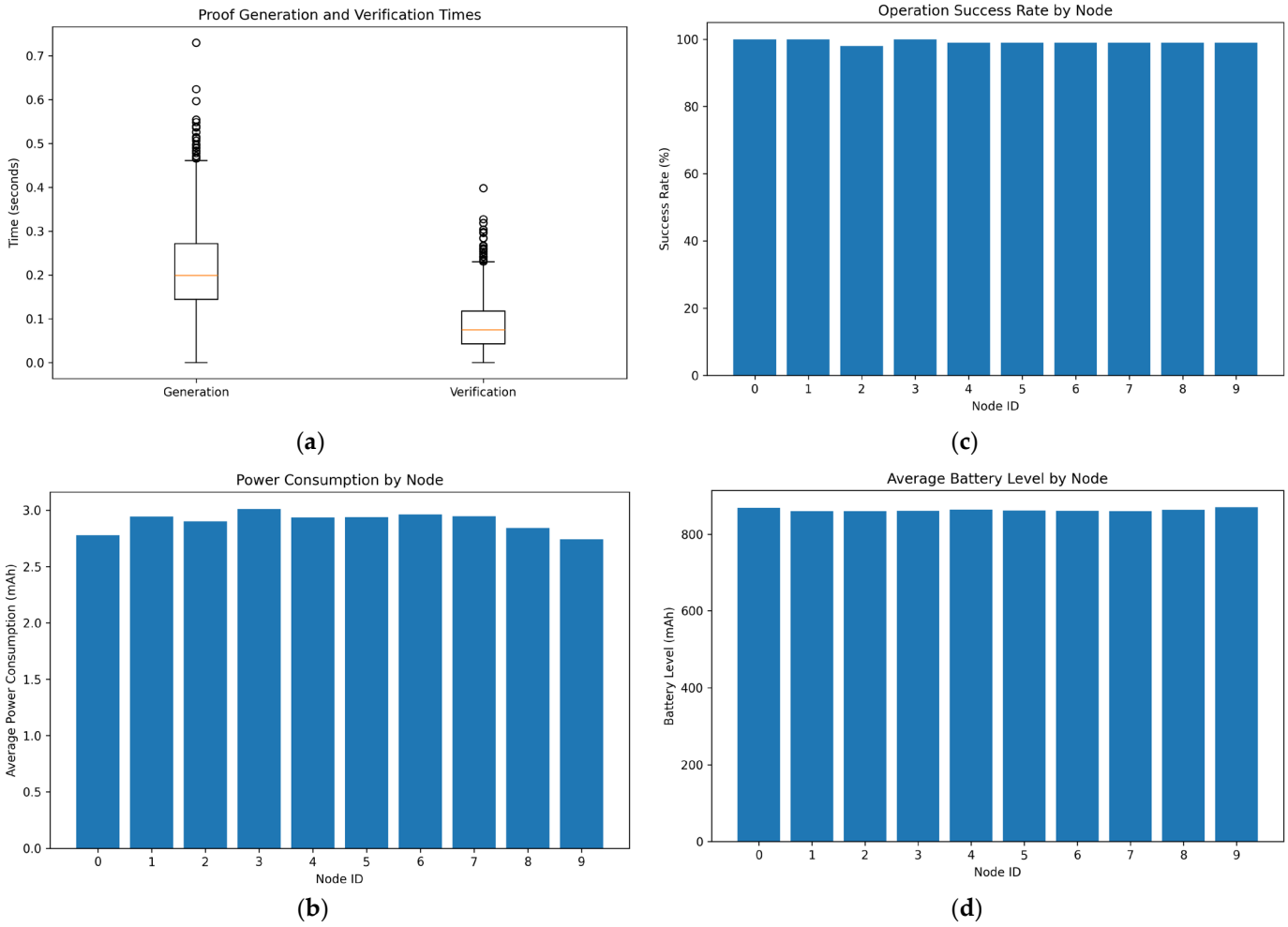


Figure 4. Performance analysis of the OR-aggregation protocol in a simulated network. (**a**) Proof generation and verification times showing distribution and outliers; (**b**) power consumption distribution across nodes; (**c**) success rate demonstrating operational reliability; (**d**) battery level monitoring indicating resource efficiency.

The simulation results demonstrate several important characteristics of our approach. Time efficiency analysis shows consistent performance with a median proof generation time of 0.214 s, while verification operations are significantly faster, averaging 0.087 s. Occasional performance variations manifested as outliers in our measurements can be attributed to simulated network conditions.

Power consumption across nodes maintains remarkable stability at approximately 2.90 mAh, with minimal variation between nodes (2.7–3.0 mAh). This consistent power consumption pattern enables reliable battery life estimation. The protocol demonstrates exceptional operational reliability, achieving a 99.20% success rate across all nodes, with uniform distribution indicating consistent performance throughout the network. The minimal failure rate of 0.8% primarily stems from simulated packet losses.

Resource efficiency metrics reveal that battery levels consistently remain above 800 mAh after all operations, with uniform consumption across nodes suggesting balanced network load. The minimal battery impact indicates strong suitability for long-term deployment. These results, combined with the constant proof size of 160 bytes and sub-second processing times, establish our OR-aggregation approach as particularly well-suited for blockchain-based sensor networks.

The constant proof size of 160 bytes, combined with sub-second processing times and minimal power requirements, makes the protocol particularly attractive for resource-

constrained IoT environments. The high success rate (99.20%) further confirms the protocol's resilience to network variations and its suitability for real-world deployment.

8. Discussion

The comparative analysis of our OR-aggregation approach against existing methods such as Patricia–Merkle trees, Verkle trees, and STARKs reveals significant implications, particularly for IoT and Wireless Sensor and Actuator Networks (WSANs). This section elaborates on these implications and discusses the broader impact of our method in various application scenarios.

8.1. Implications for IoT and WSANs

1. Bandwidth Efficiency: In IoT and WSAN environments, where bandwidth is often severely constrained, our OR-aggregation method offers a substantial advantage. The constant proof size of 160 bytes, regardless of the set size, provides a predictable and manageable data overhead. This characteristic is particularly crucial in large-scale deployments, such as smart cities or industrial IoT, where millions of devices may need to prove set membership regularly. For instance, in a scenario with 1 million devices, our method would require only 160 bytes per proof, compared to approximately 1.7 KB for Patricia–Merkle proofs or 240 bytes for Verkle trees. This reduction in data transfer can lead to significant bandwidth savings, potentially enabling more frequent updates or allowing for more devices on the same network infrastructure.
2. Energy Efficiency: Energy consumption is a critical factor in many IoT and WSAN applications, especially for battery-powered devices. The reduced data transmission requirements of our method directly translate to lower energy consumption. Consider a sensor network where each node needs to prove its membership daily. Over a year, the cumulative energy savings from transmitting smaller proofs could extend the battery life of devices by weeks or even months, depending on the specific hardware and network conditions. This extended operational life can significantly reduce maintenance costs and improve the overall reliability of the network.
3. Scalability: The constant proof size and verification time of our approach provide exceptional scalability. As IoT networks grow to encompass billions of devices, traditional methods like Patricia–Merkle trees would see logarithmic growth in proof sizes and verification times. In contrast, our method maintains its efficiency regardless of network size. This scalability is particularly valuable in rapidly expanding IoT ecosystems, where the ability to accommodate growth without degrading performance is crucial.
4. Real-time Applications: Many IoT and WSAN applications, such as industrial control systems or traffic management in smart cities, require real-time or near-real-time responses. The constant and fast verification time of our method (0.012125 s) ensures predictable performance, critical for time-sensitive applications. This consistency allows for more reliable system design and can enable new classes of applications that were previously infeasible due to verification time constraints.
5. Edge Computing Compatibility: The computational characteristics of our approach make it highly suitable for edge computing scenarios in IoT. With less powerful processors often found in edge devices, the ability to perform quick verifications without requiring significant computational resources is a major advantage. This compatibility with edge computing can reduce latency and bandwidth usage by allowing verifications to occur closer to the data source.
6. Security and Privacy: While our method primarily focuses on efficiency, it maintains the zero-knowledge property, ensuring that no information about the set elements is leaked during the proof process. This feature is particularly important in IoT scenarios where data privacy is crucial, such as in healthcare or smart home applications.

In dynamic IoT environments, proof generation frequency must balance responsiveness with resource constraints. Our experimental results suggest an optimal generation

rate of 2.85 Hz for typical sensor networks, allowing real-time verification while preventing computational bottlenecks. For large sets, our parallel processing optimization reduces generation time by a factor of n/log(n) compared to sequential processing, though this introduces additional security considerations. Specifically, the parallel generation of commitments must ensure proper synchronization to prevent race conditions and timing attacks. Our implementation mitigates these risks through atomic commitment operations and secure random number generation for each parallel thread.

The scalability of batch verification in our approach demonstrates linear growth in total processing time while maintaining constant time per proof. For example, verifying 10,000 proofs takes approximately 870 s total, with each proof consistently requiring 0.087 s regardless of batch size. This predictable scaling makes our approach particularly suitable for large-scale IoT deployments where batch verification is common.

8.2. Broader Impact and Future Directions

1. Blockchain Light Clients: Our method could revolutionize the design of blockchain light clients, particularly for IoT devices. The constant-size proofs allow for efficient state verification without the need to store or process the entire blockchain. This could enable a new class of ultra-light blockchain clients suitable for resource-constrained IoT devices.
2. Access Control Systems: In large-scale distributed systems, our method could significantly improve the efficiency of access control mechanisms. The ability to quickly verify membership in access control lists without substantial data transfer could enhance both the speed and security of authentication processes.
3. Supply Chain Management: In IoT-enabled supply chain systems, our approach could facilitate rapid verification of product authenticity or tracking status. The efficient proofs could allow for more frequent checks and updates, improving overall supply chain visibility and integrity.
4. Smart Cities: The scalability of our method makes it particularly suitable for smart city applications. From traffic management systems verifying vehicle identities to smart grids authenticating individual power consumption reports, the efficient proofs could enable more dynamic and responsive urban systems.
5. Federated Learning in IoT: In scenarios where federated learning is applied to IoT networks, our method could be used to efficiently verify the participation of devices in training rounds without revealing specific device identities.

8.3. Comparison with Advanced Zero-Knowledge Proof Systems and Post-Quantum Considerations

While our OR-aggregation approach demonstrates significant advantages for set membership proofs, it is essential to position it within the broader landscape of zero-knowledge proof systems. zk-SNARKs, zk-STARKs, and Bulletproofs each offer distinct characteristics that influence their applicability in different scenarios.

zk-SNARKs, despite requiring a trusted setup, achieve remarkable proof sizes of approximately 288 bytes, independent of computation size. However, their reliance on pairing-based cryptography not only introduces computational overhead but also raises concerns about quantum vulnerability. Our approach, while specifically tailored for set membership, achieves comparable proof sizes (160 bytes for Ed25519) without the trusted setup requirement.

Bulletproofs, which eliminate the trusted setup requirement, produce proofs that scale logarithmically with the size of the computation: $O(log(n))$. This results in smaller proofs than STARKs but larger than SNARKs. For a typical set membership proof, Bulletproofs generate proofs of approximately 672 bytes. However, their verification time scales linearly with the size of the computation, making them less suitable for large-scale IoT deployments.

zk-STARKs offer post-quantum security through their reliance on collision-resistant hash functions rather than elliptic curve cryptography. However, this advantage comes at

the cost of substantially larger proof sizes—typically 45–200 KB. While STARKs excel in proving complex computations, our method achieves better efficiency for the specific case of set membership proofs.

The post-quantum security landscape presents both challenges and opportunities for our OR-aggregation approach. While our current implementation relies on classical cryptographic assumptions (discrete logarithm and elliptic curve cryptography), the protocol's structure allows for adaptation to post-quantum primitives. Specifically:

- The underlying Schnorr protocol can be replaced with lattice-based identification schemes, such as those based on the Learning With Errors (LWE) problem.
- The commitment scheme can be modified to use hash-based commitments, maintaining security against quantum attacks while preserving the protocol's zero-knowledge properties.
- The elliptic curve operations can be substituted with isogeny-based cryptography, though this would require careful consideration of performance implications.

A preliminary analysis of adapting our protocol to post-quantum security through LWE-based commitments considered the following factors:

- The basic Schnorr commitment in our protocol requires 32 bytes for the group element. Its LWE-based replacement would require approximately 800 bytes, based on the recommended parameters for a 128-bit post-quantum security level ($n = 512$, $q \approx 2^{23}$).
- The challenge size would remain relatively small at 32 bytes, as it represents a random seed that must be unpredictable but not quantum-resistant.
- The response component would increase from 32 bytes to approximately 400 bytes due to the larger size of LWE-based responses.

These estimates suggest that a post-quantum variant of our protocol would require approximately 1.2 KB total (800B + 32B + 400B = 1232B) for a complete proof. The verification process would require additional matrix-vector multiplications instead of simple group operations, resulting in an estimated 2.5× increase in verification time due to the computational overhead of LWE operations. However, this verification time would still remain constant regardless of set size, preserving one of the key advantages of our approach.

This analysis indicates that while adapting our protocol to post-quantum security would increase resource requirements, it would still maintain significantly better efficiency than post-quantum STARKs for set membership applications. The proof size of 1.2 KB represents only a fraction of typical post-quantum STARK proof sizes (45–200 KB), while preserving our protocol's valuable property of constant-size proofs independent of set cardinality.

8.4. Limitations and Future Work

While our OR-aggregation method offers significant advantages, it is important to acknowledge its limitations:

1. Proof Generation Time: The proof generation time, while constant, is longer than some competing methods for smaller set sizes. Future work could focus on optimizing this aspect, potentially through parallelization or specialized hardware acceleration.
2. Cryptographic Assumptions: The security of our method relies on specific cryptographic assumptions. Ongoing research should continually assess these assumptions against advances in cryptanalysis and quantum computing.
3. Complex Computation Proofs: Unlike STARKs, our method is specialized for set membership proofs and does not support proofs of arbitrary computations. Future research could explore ways to extend the approach to more complex proof scenarios while maintaining its efficiency benefits.

In conclusion, our OR-aggregation method for zero-knowledge set membership proofs offers a compelling solution for IoT and WSAN applications, particularly in scenarios involving large-scale, bandwidth-constrained, or energy-sensitive environments. Its constant proof size and verification time provide a level of efficiency and scalability that is

unmatched by existing methods. As IoT and WSAN technologies continue to evolve and proliferate, the importance of efficient cryptographic proofs will only grow. Our approach represents a significant step forward in addressing these emerging needs, potentially enabling new applications and improving the performance of existing systems across a wide range of domains.

9. Conclusions

This study introduces a novel OR-aggregation approach for zero-knowledge set membership proofs, specifically designed for blockchain-based sensor networks. Our method addresses critical challenges in scalability, privacy, and efficiency that have hindered the widespread adoption of blockchain technology in IoT ecosystems. By leveraging the mathematical properties of both prime-order groups and elliptic curve cryptography, we achieve constant-size proofs and verification times, independent of set size. This represents a significant improvement over traditional methods such as Merkle trees and accumulator-based approaches.

Our comprehensive evaluation demonstrates the superiority of our approach in terms of proof size, generation time, and verification efficiency, particularly for large-scale deployments. The integration strategies we propose for prominent blockchain platforms pave the way for practical implementation in real-world scenarios. While our work makes substantial progress in enabling efficient and privacy-preserving set membership proofs for IoT applications, future research should focus on further optimizing performance for ultra-resource-constrained devices and exploring the implications of quantum computing on the long-term security of our approach.

Author Contributions: Conceptualization, O.K.; methodology, O.K.; data curation, E.F.; formal analysis, M.A.; investigation, K.K.; writing—original draft preparation, K.K.; writing—review and editing, O.K.; supervision, E.F.; funding acquisition, M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 101007820—TRUST. This publication reflects only the author's view, and the REA is not responsible for any use that may be made of the information it contains. This research was funded by the European Union—NextGenerationEU under the Italian Ministry of University and Research (MIUR), National Innovation Ecosystem Grant ECS00000041-VITALITY-CUP D83C22000710005.

Data Availability Statement: The original contributions presented in the study are included in the article; the datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

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

References

1. Ma, N.; Waegel, A.; Hakkarainen, M.; Braham, W.W.; Glass, L.; Aviv, D. Blockchain + IoT Sensor Network to Measure, Evaluate and Incentivize Personal Environmental Accounting and Efficient Energy Use in Indoor Spaces. *Appl. Energy* **2023**, *332*, 120443. [CrossRef]
2. Chen, Y.; Yang, X.; Li, T.; Ren, Y.; Long, Y. A Blockchain-Empowered Authentication Scheme for Worm Detection in Wireless Sensor Network. *Digit. Commun. Netw.* **2024**, *10*, 265–272. [CrossRef]
3. Dwivedi, S.K.; Amin, R.; Vollala, S. Design of Secured Blockchain Based Decentralized Authentication Protocol for Sensor Networks with Auditing and Accountability. *Comput. Commun.* **2023**, *197*, 124–140. [CrossRef]
4. Godawatte, K.; Branch, P.; But, J. Use of Blockchain in Health Sensor Networks to Secure Information Integrity and Accountability. *Procedia Comput. Sci.* **2022**, *210*, 124–132. [CrossRef]
5. Hanggoro, D.; Windiatmaja, J.H.; Muis, A.; Sari, R.F.; Pournaras, E. Energy-Aware Proof-of-Authority: Blockchain Consensus for Clustered Wireless Sensor Network. *Blockchain Res. Appl.* **2024**, *5*, 100211. [CrossRef]
6. Patel, N.; Arora, A.; Aggarwal, M. Evaluating Simulation Tools for Securing Sensor Data with Blockchain: A Comprehensive Analysis. *Meas. Sens.* **2024**, *33*, 101233. [CrossRef]
7. Faheem, M.; Al-Khasawneh, M.A.; Khan, A.A.; Madni, S.H.H. Cyberattack Patterns in Blockchain-Based Communication Networks for Distributed Renewable Energy Systems: A Study on Big Datasets. *Data Brief* **2024**, *53*, 110212. [CrossRef]

8. Hofstadler, C.; Verron, T. Short Proofs of Ideal Membership. *J. Symb. Comput.* **2024**, *125*, 102325. [CrossRef]
9. Liu, H.; Luo, X.; Liu, H.; Xia, X. Merkle Tree: A Fundamental Component of Blockchains. In Proceedings of the 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 23–26 September 2021; pp. 556–561.
10. Jeon, K.; Lee, J.; Kim, B.; Kim, J.J. Hardware Accelerated Reusable Merkle Tree Generation for Bitcoin Blockchain Headers. *IEEE Comput. Archit. Lett.* **2023**, *22*, 69–72. [CrossRef]
11. Ozcelik, I.; Medury, S.; Broaddus, J.; Skjellum, A. An Overview of Cryptographic Accumulators. In Proceedings of the 7th International Conference on Information Systems Security and Privacy ICISSP—Volume 1, 2021, Virtual, 11–13 February 2021; pp. 661–669.
12. ZK Whiteboard Sessions. ZK Whiteboard Sessions. Available online: https://zkhack.dev/whiteboard (accessed on 5 October 2024).
13. Wu, J.; Yuan, L.; Xie, T.; Dai, H. A Sharding Blockchain Protocol for Enhanced Scalability and Performance Optimization through Account Transaction Reconfiguration. *J. King Saud Univ. Comput. Inf. Sci.* **2024**, *36*, 102184. [CrossRef]
14. Jung, S.; Yoo, Y.; Yang, G.; Yoo, C. Prediction of Permissioned Blockchain Performance for Resource Scaling Configurations. *ICT Express* 2024, *in press*. [CrossRef]
15. Haddad, Z. Enhancing Privacy and Security in 5G Networks with an Anonymous Handover Protocol Based on Blockchain and Zero Knowledge Proof. *Comput. Netw.* **2024**, *250*, 110544. [CrossRef]
16. Zhou, L.; Diro, A.; Saini, A.; Kaisar, S.; Hiep, P.C. Leveraging Zero Knowledge Proofs for Blockchain-Based Identity Sharing: A Survey of Advancements, Challenges and Opportunities. *J. Inf. Secur. Appl.* **2024**, *80*, 103678. [CrossRef]
17. Li, Y.; Wang, J.; Zhang, H. A Survey of State-of-the-Art Sharding Blockchains: Models, Components, and Attack Surfaces. *J. Netw. Comput. Appl.* **2023**, *217*, 103686. [CrossRef]
18. Nasir, M.H.; Arshad, J.; Khan, M.M.; Fatima, M.; Salah, K.; Jayaraman, R. Scalable Blockchains—A Systematic Review. *Future Gener. Comput. Syst.* **2022**, *126*, 136–162. [CrossRef]
19. Cai, Q.; Zhang, Z.; Fu, M. Distributed Computations for Large-Scale Networked Systems Using Belief Propagation. *J. Autom. Intell.* **2023**, *2*, 61–69. [CrossRef]
20. Farina, M.; Rocca, M. A Novel Distributed Algorithm for Estimation and Control of Large-Scale Systems. *Eur. J. Control* **2023**, *72*, 100820. [CrossRef]
21. Rawhouser, H.; Webb, J.W.; Rodrigues, J.; Waldron, T.L.; Kumaraswamy, A.; Amankwah-Amoah, J.; Grady, A. Scaling, Blockchain Technology, and Entrepreneurial Opportunities in Developing Countries. *J. Bus. Ventur. Insights* **2022**, *18*, e00325. [CrossRef]
22. Mlika, F.; Karoui, W.; Romdhane, L.B. Blockchain Solutions for Trustworthy Decentralization in Social Networks. *Comput. Netw.* **2024**, *244*, 110336. [CrossRef]
23. Woltering, L.; Valencia Leñero, E.M.; Boa-Alvarado, M.; Van Loon, J.; Ubels, J.; Leeuwis, C. Supporting a Systems Approach to Scaling for All; Insights from Using the Scaling Scan Tool. *Agric. Syst.* **2024**, *217*, 103927. [CrossRef]
24. Zhang, H.; Guo, T.; Tian, W.; Ma, H. Learning-Driven Hybrid Scaling for Multi-Type Services in Cloud. *J. Parallel Distrib. Comput.* **2024**, *189*, 104880. [CrossRef]
25. Bin Hasan, K.M.; Sajid, M.; Lapina, M.A.; Shahid, M.; Kotecha, K. Blockchain Technology Meets 6 G Wireless Networks: A Systematic Survey. *Alex. Eng. J.* **2024**, *92*, 199–220. [CrossRef]
26. Li, M.; Peng, C.; Han, L.; Feng, Q.; Luo, M. High-Speed Batch Verification for Discrete-Logarithm-Based Signatures via Multi-Scalar Multiplication Algorithm. *J. Inf. Secur. Appl.* **2024**, *87*, 103898. [CrossRef]
27. Krishnan, D.; Fu, X. Towards Zero Knowledge Argument for Double Discrete Logarithm with Constant Cost. *Theor. Comput. Sci.* **2024**, *1018*, 114799. [CrossRef]
28. Ullah, S.; Zheng, J.; Din, N.; Hussain, M.T.; Ullah, F.; Yousaf, M. Elliptic Curve Cryptography; Applications, Challenges, Recent Advances, and Future Trends: A Comprehensive Survey. *Comput. Sci. Rev.* **2023**, *47*, 100530. [CrossRef]
29. Adeniyi, A.E.; Jimoh, R.G.; Awotunde, J.B. A Systematic Review on Elliptic Curve Cryptography Algorithm for Internet of Things: Categorization, Application Areas, and Security. *Comput. Electr. Eng.* **2024**, *118*, 109330. [CrossRef]
30. Aggarwal, S.; Kumar, N. Digital Signatures☆. In *Advances in Computers*; Aggarwal, S., Kumar, N., Raj, P., Eds.; The Blockchain Technology for Secure and Smart Applications across Industry Verticals; Elsevier: Amsterdam, The Netherlands, 2021; Volume 121, pp. 95–107.
31. Tiwari, A. Chapter 14—Cryptography in Blockchain. In *Distributed Computing to Blockchain*; Pandey, R., Goundar, S., Fatima, S., Eds.; Academic Press: Cambridge, MA, USA, 2023; pp. 251–265, ISBN 978-0-323-96146-2.
32. Bartoli, C.; Cascudo, I. On Sigma-Protocols and (Packed) Black-Box Secret Sharing Schemes. In *IACR International Conference on Public-Key Cryptography*; Springer Nature: Cham, Switzerland, 2023.
33. Ciampi, M.; Persiano, G.; Scafuro, A.; Siniscalchi, L.; Visconti, I. Improved OR-Composition of Sigma-Protocols. In *Proceedings of the Theory of Cryptography*; Kushilevitz, E., Malkin, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; pp. 112–141.
34. Zhang, M.; Chen, Y.; Yao, C.; Wang, Z. Sigma Protocols from Verifiable Secret Sharing and Their Applications. In *International Conference on the Theory and Application of Cryptology and Information Security*; Springer Nature: Singapore, 2023.
35. Deng, S.; Du, B. zkTree: A Zero-Knowledge Recursion Tree with ZKP Membership Proofs. *Cryptol. ePrint Arch.* 2023, *preprint*. Available online: https://eprint.iacr.org/2023/208 (accessed on 5 October 2024).

36. Kuznetsov, O.; Rusnak, A.; Yezhov, A.; Kanonik, D.; Kuznetsova, K.; Domin, O. Efficient and Universal Merkle Tree Inclusion Proofs via OR Aggregation. *Cryptography* **2024**, *8*, 28. [CrossRef]
37. Kuznetsov, O.; Rusnak, A.; Yezhov, A.; Kanonik, D.; Kuznetsova, K.; Karashchuk, S. Enhanced Security and Efficiency in Blockchain With Aggregated Zero-Knowledge Proof Mechanisms. *IEEE Access* **2024**, *12*, 49228–49248. [CrossRef]
38. Kuznetsov, O. Google Colab: Ed25519SetMembership for IoT. 2024. Available online: https://colab.research.google.com/drive/1wX6HoJk3MDWCkRnv7ee2sBOiyznRfYU1?usp=sharing (accessed on 6 November 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.