International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V74I10P101

Volume 74 Issue 10, 1-9, October 2026
© 2026 Seventh Sense Research Group®

Original Article

A Merkle-Based Aggregation Framework for Cost-Efficient Blockchain Document Verification

Rocendo F. Astillero

Polytechnic University of the Philippines.

Corresponding Author : rocendoastillero@gmail.com

Received: 16 August 2026 Revised: 20 September 2026 Accepted: 08 October 2026 Published: 29 October 2026

Abstract *- Blockchain-based document verification can establish a tamper-evident record without placing the complete document on-chain. However, anchoring every document hash in a separate transaction creates transaction-count and fee overhead that grows with the number of registered documents. This study proposes a Merkle-Based Aggregation Framework that combines off-chain document storage, cryptographic hashing, Merkle-root batching, blockchain anchoring, and individual inclusion-proof verification. The framework changes the unit of blockchain commitment from an individual document to a batch while retaining document-level verification. Its principal contribution is a framework and evaluation method for determining when aggregation is economically useful, rather than a claim that Merkle trees themselves are novel. The framework defines transaction count, gas consumption, transaction fee, amortized fee per document, verification latency, proof size, throughput, and break-even batch size as evaluation measures. An analytical model shows that N documents require N anchoring transactions under per-document registration, whereas one root transaction suffices for a complete batch under the proposed architecture. The model also shows that cost efficiency depends on root-transaction gas growth and the effective gas price; therefore, transaction reduction must not be treated as equivalent to proportional fee reduction. I use existing experimental findings on Merkle batching and blockchain-based document verification to position the framework and define the benchmark conditions. Empirical gas and latency measurements remain necessary before making numerical performance claims.*

Keywords *- Batch processing, Blockchain anchoring, Cryptographic hashing, Decentralized storage, Document integrity, inclusion proof.*

1. Introduction

Digital academic, legal, financial, and institutional records increasingly require mechanisms that can establish document integrity and support later verification. Blockchain can provide an append-only commitment layer, while off-chain systems can retain the document content. IPFS is relevant to this architecture because it uses content addressing and distributed storage rather than requiring complete documents to be placed directly on a blockchain [1]. Ethereum provides a programmable transaction environment in which execution and storage operations incur gas costs [2].

A common blockchain document-verification pattern computes a document's cryptographic digest and records it on-chain. The approach is straightforward for small workloads, but the number of blockchain transactions increases with the number of documents. If N documents are independently anchored, N anchoring transactions are required. For an institution that registers documents in batches, this repeated transaction overhead becomes a scalability and cost concern. Existing studies have demonstrated that Merkle structures can represent multiple verifiable records through a compact root commitment. Saito and Watanabe demonstrated blockchain-based verification of multiple document structures using a Merkle tree and inclusion proofs [3]. Patil et al. later presented MerkleDoc, which applies page-level hashing, Merkle trees, public blockchain anchoring, and issuer signatures for document verification [4]. More recently, Andrian and Wiputra evaluated Merkle-tree batching with IPFS for blockchain-based micro-credentials and reported substantially more stable gas consumption for batched issuance than for individual issuance [5]. These studies establish the technical feasibility of Merkle-based aggregation, but they address different workloads and evaluation objectives. This study, therefore, does not address whether a Merkle tree can be used for document verification. That capability is established. The problem is how a document-verification framework can use aggregation as an explicit cost-control mechanism while preserving individual document verification and providing a repeatable method for selecting batch sizes. In particular, the framework must distinguish transaction-count reduction from actual gas and monetary-fee reduction because a larger root transaction can consume more gas as batch-related data and contract processing increase.


This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

The research gap is the absence of a document-centric evaluation framework that jointly treats batch size, root-transaction gas, effective gas price, amortized document cost, verification overhead, and break-even conditions as connected variables. Existing work provides important building blocks: Merkle-based document verification [3], privacy-preserving document verification [4], and empirical batching for micro-credentials [5]. The present framework connects these elements around the specific decision problem of when batch aggregation is economically justified for document registration.

The study consequently aims to design a Blockchain–IPFS document verification framework in which document hashes are aggregated into Merkle trees and only the root is anchored on-chain. The framework further defines a controlled comparison against per-document anchoring. It specifies the measurements required to determine transaction savings, gas behavior, amortized cost, verification performance, and the break-even batch size.

2. Objectives of the Study

- Design a Blockchain–IPFS document verification architecture centered on Merkle-root batch anchoring.
- Define a batch-closure mechanism using batch size and maximum waiting time, with optional cost thresholds.
- Compare transaction counts between per-document anchoring and Merkle-root batch anchoring.
- Measure gas consumption, transaction fees, and amortized fee per document across controlled batch sizes.
- Evaluate individual-document verification using Merkle inclusion proofs.
- Determine the break-even batch size under specified blockchain, contract, and gas-price conditions.
- Assess integrity, provenance, availability, confidentiality, and cost-efficiency controls within the proposed architecture.

3. Review of Related Literature

3.1. Blockchain and Off-Chain Document Integrity

Blockchain-based document systems generally avoid storing complete documents on-chain because large content increases storage and execution overhead. Instead, systems can commit a cryptographic representation to the blockchain while keeping the content in an external or decentralized storage layer. IPFS provides content-addressed storage and uses a Merkle-DAG structure to identify stored objects [1]. This separation lets the blockchain serve as an integrity and provenance layer while the document remains off-chain.

3.2. Hashing and Merkle Structures

Cryptographic hashing converts document content into a fixed-length digest. Modifying the document changes the digest, providing a basic mechanism for detecting unauthorized changes. Merkle trees extend this mechanism by recursively combining hashes until a single root represents the collection. The cryptographic literature establishes the underlying construction; therefore, the present study does not treat it as a new invention [6]. The research contribution lies in incorporating the structure into an institutional document-registration framework and evaluating it as a cost-control mechanism.

3.3. Blockchain Document Verification

Saito and Watanabe presented a method for lightweight selective disclosure in which a Merkle tree represents multiple document structures and the root is stored on a blockchain [3]. Their work demonstrates that aggregation can coexist with selective verification. Patil et al. proposed MerkleDoc, using page-level SHA-256 hashes, a Merkle tree, blockchain anchoring, and issuer signatures; the work further demonstrates individual verification through compact inclusion proofs [4]. These studies establish the feasibility of Merkle-based document verification but do not by themselves define the batch-size break-even problem targeted by this framework.

3.4. Batching and Blockchain Transaction Cost

Andrian and Wiputra examined blockchain-based micro-credential issuance using batching, Merkle trees, and IPFS [5]. Their 2026 journal study reports experiments on Sepolia and describes stable batch gas consumption around 90,487 gas units across the tested issuance workloads, together with throughput measurements. The study is especially relevant because it provides empirical evidence that batching can change the relationship between workload size and blockchain resource consumption. However, micro-credential issuance is not identical to institutional document verification: the document-level verification model, provenance fields, inclusion-proof handling, and break-even analysis require separate treatment.

3.5. Gas, Transaction Count, and Amortized Cost

Ethereum transaction execution is priced in gas, and the effective monetary fee depends on gas consumed and the applicable gas price [2]. Consequently, reducing the number of transactions does not automatically reduce monetary cost proportionally. A fair comparison must report at least transaction count, gas per operation, total fee, and fee per covered document. This distinction is central to the present framework.

3.6. Research Gap

The literature establishes three relevant facts. First, blockchain can provide an integrity commitment without storing the complete document on-chain [1], [2]. Second, Merkle structures can compress many document-related hashes into a root while retaining individual inclusion proofs [3], [4], [6]. Third, batching can reduce the transaction overhead associated with large-scale blockchain issuance [5]. What remains insufficiently specified for the document-

verification workload is a repeatable framework that connects these mechanisms to batch-size selection, root-transaction gas growth, amortized cost, verification overhead, and a measurable break-even condition. The proposed framework addresses this methodological gap.

3.7. Comparison with Existing Research

Table 1. Comparison research table

Study	Application	Aggregation / Storage	Primary contribution	Remaining issue is addressed here
Saito & Watanabe [3]	Verifiable documents	Merkle tree + blockchain	Selective disclosure and proof of existence	Does not center batch-size cost optimization for institutional registration
Patil et al. [4]	Document verification	Page hashes + Merkle tree + blockchain	Privacy-preserving verification and issuer binding	Focuses on verification architecture rather than break-even batch economics
Andrian & Wiputra [5]	Academic micro-credentials	Merkle batching + IPFS	Empirical scalability and gas evaluation	Credential issuance differs from general institutional document verification.
Proposed framework	Institutional documents	SHA-256 + Merkle batch + IPFS + blockchain	Batch-size decision model, cost decomposition, inclusion-proof verification, controlled benchmark	Requires empirical execution to validate the analytical model

4. Methodology

4.1. Research Design

The study follows a design-and-evaluation approach. The study defines a baseline registration path and an aggregated registration path using the same document corpus, hashing algorithm, smart-contract environment, machine configuration, and measurement procedure. The baseline anchors each document hash independently. The proposed path groups document hashes into batches, computes a Merkle root, and anchors one root per completed batch. The evaluation is intended to isolate the effect of aggregation rather than differences in hashing, storage, or application logic.

4.2. System Architecture

Proposed Blockchain–IPFS document verification architecture. The three-tier workflow comprises: (1) Off-Chain Storage & Hashing Layer: Submitted documents are assigned metadata and hashed using SHA-256; document content is uploaded to IPFS to obtain a Content Identifier (CID), maintaining privacy and avoiding costly on-chain data storage12. (2) Aggregation Layer: Document hashes are grouped into a pending batch governed by size ($Bmax$) and waiting time ($Tmax$) constraints, where a balanced binary Merkle tree is constructed to derive a single 256-bit Merkle root. (3) On-Chain Commitment Layer: Only the Merkle root, batch ID, and timestamp are anchored on-chain via smart contract transactions1. Verification recomputes the target document digest and uses an $0(log_2\ B)$ Merkle inclusion proof to reconstruct the root, verifying it against the immutable blockchain record without exposing other documents in the batch.

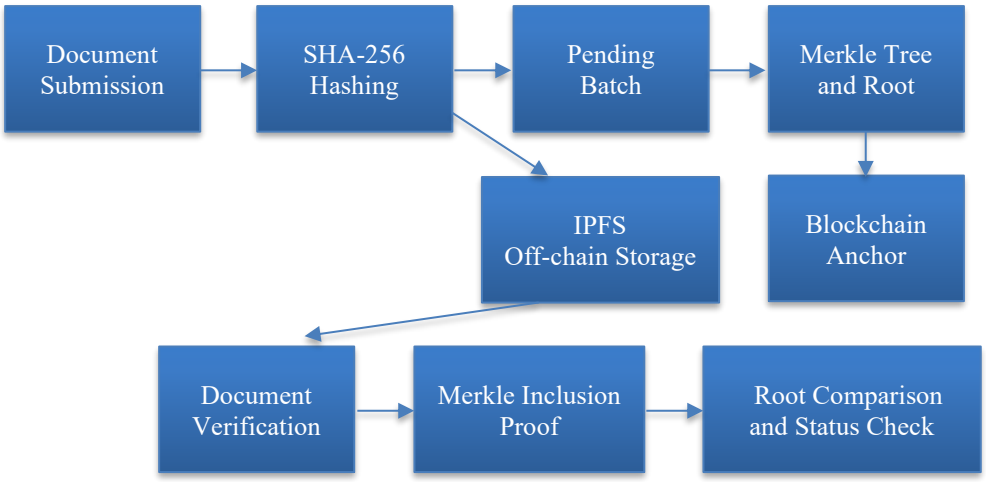


Fig. 1 Proposed Blockchain–IPFS document verification architecture

Documents are hashed before batch formation; the Merkle root and batch metadata are anchored on-chain, while document content and proof references remain off-chain. Verification recomputes the document hash, applies the inclusion proof, and compares the resulting root with the immutable blockchain commitment.

The proposed architecture combines several key technologies: blockchain, IPFS, cryptographic hashing, and Merkle-tree aggregation to create an efficient system for verifying document integrity. When registering a document, the system first processes it with a cryptographic hash function to produce a document hash. The system collects the hashes into a batch and arranges them into a Merkle tree. The resulting Merkle root is a small-sized cryptographic signature that represents everything in the batch of documents.

The proposed architecture does not store the full document on the blockchain; instead, it stores proof references and the document content off-chain using IPFS. The Merkle root is stored on the blockchain as an immutable integrity commitment. This separation enables the system to provide trusted integrity anchoring by using the blockchain and to avoid saving large documents on-chain.

At verification time, the system re-hashes the document using the same hashing algorithm. Upon receiving the Merkle inclusion proof, the system reconstructs the document's Merkle root using the proof elements. The system compares the reconstructed root with the Merkle root previously recorded on the blockchain. A match means the document hash matches the registered batch; a mismatch means that the presented document or verification information doesn't match the original blockchain commitment.

The architecture provides a verification process in which document content is kept off-chain, and its integrity is anchored on-chain in a single Merkle-root commitment. The design enables batch-level verification while minimizing on-chain data and still allows individual document verification via Merkle inclusion proofs.

4.3. Batch Formation

Each submitted document is assigned an identifier and version, hashed using SHA-256, and associated with an IPFS content identifier. The resulting digest is placed in a pending batch.

A batch is closed when either the maximum batch size $Bmax$ is reached or the maximum waiting time $Tmax$ expires. An optional cost threshold can be introduced when the system has reliable gas prices and root gas estimates. The controller determines when to publish a root; it does not change the root's cryptographic meaning.

Table 2. Batch Formation

Condition	Action	Purpose
Batch size reaches $Bmax$	Compute root and anchor	Bound batch growth
Waiting time reaches $Tmax$	Compute root and anchor	Bound registration delay
Estimated amortized cost reaches target.	Close batch if configured	Support economic control
No pending documents	No transaction	Avoid empty anchors

4.4 Cost Model

For N documents registered independently, the baseline transaction count is:

$$T_baseline(N) = N$$

If documents are processed in batches of B documents, the number of root transactions is:

$$T_batch(N,B) = ceil(N / B)$$

The transaction-count reduction is therefore:

$$R_T = [1 - T_batch(N,B) / N] \times 100$$

Let g_d denote the gas used for an individual document transaction, $g_r(B)$ the gas used by a root transaction for a batch of size B, and p the applicable effective gas price. The baseline and aggregated fees can be expressed as:

$$F_baseline = N \times g_d \times p$$

$$F_batch = ceil(N / B) \times g_r(B) \times p$$

$$F_doc = F_batch / N$$

The break-even condition for a fixed batch size B is:

$$ceil(N / B) \times g_r(B) \leq N \times g_d$$

This condition is important because $g_r(B)$ should not be assumed constant. Larger batches may increase call data, contract processing, event data, or other execution requirements.

Accordingly, a reduction in transaction count is a structural result of the architecture, whereas a reduction in gas and monetary fee is an empirical result that must be measured under the selected smart contract and blockchain conditions.

4.5. Document Verification Procedure

1. Receive the document and institutional identifier.
2. Compute the SHA-256 digest.
3. Upload the document to IPFS and retain the content identifier.
4. Place the digest in the pending Merkle batch.
5. Close the batch according to the configured batch policy and compute the root.
6. Anchor the root and batch metadata in the smart contract.
7. Store the leaf position and inclusion-proof information.
8. During verification, retrieve the document, recompute its digest, and reconstruct the root using the inclusion proof.
9. Compare the reconstructed root with the blockchain-anchored root and validate document version and status metadata.
10. Return a valid result only when the digest, proof, root, and metadata checks are consistent.

4.6. Experimental Conditions

Each condition should be repeated sufficiently to estimate variation. For every experimental condition, the final report should provide sample size, mean, standard deviation, document-size distribution, smart-contract version, blockchain configuration, and gas-price conditions. If the distribution and assumptions support parametric analysis, use mean-comparison procedures; otherwise, select a non-parametric test.

Table 3. Trials

Variable	Proposed levels	Measurement
Workload	50, 100, 250, 500, 1,000 documents	Number of documents
Batch size	1, 2, 4, 8, 16, 32, 64, 128, 256, 512	Documents per root
Document size	Small, medium, large	Bytes
Anchoring mode	Individual; Merkle batch	Architecture
Blockchain	Controlled test network; optional public testnet	Network condition
Replicates	At least 5 runs per condition	n, mean, standard deviation
Gas price	Recorded per transaction	Effective price
Contract	Same version for both conditions	Contract identifier

4.7. Evaluation Metrics

Table 4. Metrics

Metric	Definition	Interpretation
Transaction count	Number of anchoring transactions	Structural reduction from aggregation
Gas used	Gas units consumed by anchoring	Execution overhead
Transaction fee	Gas used × effective gas price	Monetary cost under test conditions
Amortized fee/document	Total batch fee ÷ documents covered	Economic effect of aggregation
Proof size	Bytes or hashes in inclusion proof	Verification overhead
Verification latency	Time from request to verification result	Usability
Throughput	Documents processed per unit time	Scalability
Integrity detection	Correctly detected modified documents	Security property
Break-even batch size	Smallest batch size meeting the cost condition	Operational decision point

Table 5. Count Effect

Workload	Batch size	Baseline transactions	Batch transactions	Analytical reduction
1,000	10	1,000	100	90.0%
1,000	50	1,000	20	98.0%
1,000	100	1,000	10	99.0%
500	50	500	10	98.0%
100	10	100	10	90.0%

5. Analytical Results and Discussion

Because the submitted manuscript does not contain a completed experimental dataset, this section separates analytical results that follow directly from the proposed model from empirical claims that require execution. This distinction prevents presenting theoretical relationships as measured performance.

5.1. Transaction-Count Effect

For a workload of N documents and a uniform batch size B, the proposed architecture requires ceil(N/B) root

transactions. For example, a workload of 1,000 documents requires 1,000 anchoring transactions under the baseline and 100 root transactions when B=10, 20 root transactions when B=50, or 10 root transactions when B=100. These values are analytical consequences of the batching policy, not measurements from a blockchain network.

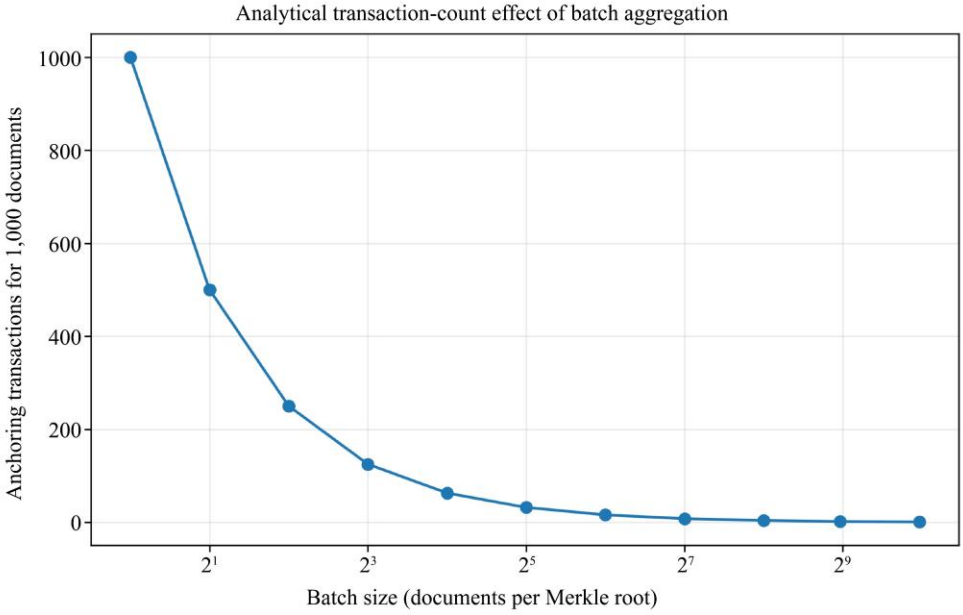


Fig. 2 Analytical relationship between batch size (B) and anchoring transaction count for a 1,000-document workload

The asymptotic curve demonstrates the structural reduction in required blockchain commitments, decreasing transaction count from $N = 1{,}000$ independent baseline transactions to N/B batch transactions. Aggregation yields a 90% transaction reduction at $B = 10$, reaching 99% at $B = 100$. Note that this curve depicts structural transaction count only; monetary fee reduction requires empirical gas measurement $(Gr(B))$ to account for potential root-transaction calldata and storage overhead.

5.2. Amortized Cost and Break-Even Behavior

Transaction count alone cannot capture the economic effect. If the root transaction consumes more gas as the batch grows, the decrease in transaction count may be accompanied by higher root-transaction execution cost. The relevant quantity is therefore the total fee divided by the number of covered documents. The break-even point occurs at the smallest batch size for which the measured batch fee is no greater than the corresponding baseline fee. This formulation produces a practical decision rule. If $g_r(B)/B$ is below the measured individual anchoring gas g_d, aggregation reduces the gas burden per document. $If\ g_r(B)/B$ approaches or exceeds g_d, increasing the batch size further provides little or no economic benefit. Thus, the expected relationship is a declining amortized cost curve with a possible leveling region rather than an assumption of unlimited savings.

5.3. Individual Verification Remains Possible

Aggregation does not require losing document-level verification. A verifier can recompute the document hash and use the associated sibling hashes to reconstruct the root. For a balanced binary Merkle tree, the proof path grows logarithmically with the number of leaves, approximately $O(log2\ B)$, rather than requiring inspection of all documents in the batch. This property allows the proposed framework to separate the frequency of blockchain anchoring from the granularity of document verification.

5.4. Comparison with Reported Research

The proposed framework differs from earlier work primarily in its evaluation question. Saito and Watanabe established that multiple verifiable document structures can be represented by a Merkle tree and verified against a blockchain root [3]. MerkleDoc further demonstrated a document-verification architecture combining page-level hashing, Merkle roots, blockchain anchoring, and issuer signatures [4]. Andrian and Wiputra provided experimental evidence for batching and IPFS in academic micro-credential issuance, including gas, throughput, and execution-time measurements [5]. The present work brings these established mechanisms into a single document-registration framework in which batch size is treated as an explicit cost variable and the break-even condition is part of the evaluation design.

The framework should therefore not be described as outperforming state-of-the-art techniques until the same workload and blockchain conditions have been executed and measured. The defensible contribution at the revision stage is the framework, cost model, controlled comparison protocol, and security-preserving verification design. Once

measurements are available, the Results section should report measured gas, fees, latency, throughput, proof sizes, and statistical variation against the baseline, then quantify the observed improvement.

5.5. Security and Operational Analysis

Threat/concern	Control	Expected property
Document tampering	SHA-256 digest + inclusion proof	Integrity
False batch membership	Anchored root + inclusion proof	Membership authenticity
Unauthorized version replacement	Version identifier + immutable batch record	Provenance
Metadata alteration	Controlled/signed smart-contract fields	Record integrity
IPFS unavailability	Pinning and redundant storage	Availability
Unauthorized disclosure	Encryption and access control	Confidentiality
Excessive anchoring cost	Batch aggregation + break-even policy	Cost efficiency

The security model has an important boundary: a matching hash establishes consistency with the committed content, but does not, by itself, establish the issuer's identity or authority. Where institutional authenticity is required, integrate issuer signatures or controlled issuer identities. Similarly, IPFS content addressing does not guarantee permanent availability; pinning and redundancy remain operational requirements [1].

6. Contributions and Novelty Position

The principal contribution is a framework-level integration of established cryptographic and decentralized-storage mechanisms into a cost-oriented document-verification workflow.

First, the framework defines Merkle aggregation as the unit of blockchain commitment: multiple document hashes are represented by one root, reducing repeated anchoring operations while retaining individual inclusion proofs.

Second, the framework separates transaction-count reduction, gas reduction, monetary-fee reduction, and amortized cost per document. This prevents the common analytical error of treating fewer transactions as direct evidence of proportional monetary savings.

Third, the framework introduces break-even batch analysis. The framework does not assume the largest possible batch size; it determines the appropriate batch size from measured root-transaction gas, individual anchoring gas, effective gas price, and operational constraints such as waiting time.

Fourth, the framework integrates document provenance fields, IPFS content identifiers, batch identifiers, version information, and inclusion-proof references so that cost optimization does not remove document-level verification.
Fifth, the framework specifies a reproducible benchmark in which the baseline and proposed systems use the same document corpus, hashing algorithm, contract version, blockchain condition, hardware, and measurement procedure.

7. Limitations and Required Empirical Validation

The principal limitation of the current revision is the absence of a completed experimental dataset in the source manuscript. Consequently, the manuscript does not claim numerical gas savings, fee reductions, latency improvements, throughput gains, or an observed break-even batch size.

Actual root-transaction gas can vary with contract implementation, calldata size, event emission, storage writes, and batch metadata. Public-testnet fees can also vary with network conditions and effective gas price. The proposed model, therefore, requires direct measurement.

Batching can introduce waiting time before anchoring a root. A large batch may reduce amortized cost but increase registration delay or exceed practical block and contract limits. The final empirical study should therefore evaluate cost and latency together.

IPFS availability depends on pinning and replication, and confidentiality requires encryption and access control when documents contain sensitive information.

Before final submission, populate the experimental results section only with measurements generated by the implemented system. The final paper should report mean, standard deviation, sample size, test conditions, and appropriate statistical tests. Obtain a Turnitin report after all revisions; manuscript editing alone cannot guarantee the similarity threshold.

8. Conclusion

The revised study positions Merkle aggregation as a framework-level cost and scalability mechanism for Blockchain–IPFS document verification rather than as a new cryptographic construction. The proposed architecture changes the anchoring unit from an individual document to a batch while preserving individual verification through Merkle inclusion proofs. The analytical model establishes a direct relationship between workload size, batch size, transaction

count, root-transaction gas, and amortized document cost. It also establishes why break-even analysis is necessary: the reduction from N individual transactions to approximately N/B batch transactions is structural, but the monetary benefit depends on the measured gas behavior of the root transaction and the effective gas price.

The study positions Merkle aggregation as a framework-level cost and scalability mechanism for Blockchain–IPFS document verification rather than as a new cryptographic construction. The proposed architecture changes the anchoring unit from an individual document to a batch while preserving individual verification through Merkle inclusion proofs. The analytical model establishes a direct relationship between workload size, batch size, transaction count, root-transaction gas, and amortized document cost. It also explains why break-even analysis is necessary: the reduction from N individual transactions to approximately N/B batch transactions is structural, but the monetary benefit depends on the root transaction's measured gas behavior and the effective gas price. The framework supports empirical comparison between per-document and batch anchoring under identical conditions. The final validation should measure gas, fees, latency, throughput, proof size, integrity detection, and break-even batch size using repeated experimental runs. With those measurements, the study can determine the practical operating range in which Merkle-based aggregation reduces blockchain anchoring cost without sacrificing document-level verification or provenance.

Author Contribution Statement

Rocendo F. Astillero conceived the study, designed the framework and evaluation methodology, developed the proposed architecture and cost model, and prepared the manuscript.

Acknowledgment

The author acknowledges the academic and institutional support that helped develop the research concept.

Funding

This research received no external funding.

References

[1] Juan Benet, "IPFS - Content Addressed, Versioned, P2P File System," *arXiv*, pp. 1-11, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[2] Gavin Wood, "Ethereum: A Secure Decentralized Generalized Transaction Ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1-32, 2014. [Google Scholar]

[3] Kenji Saito, and Satoki Watanabe, "Lightweight Selective Disclosure for Verifiable Documents on Blockchain," *ICT Express*, vol. 7, no. 3, pp. 290-294, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Akshat Patil et al., "MerkleDoc: A Privacy-Preserving Blockchain-based Document Verification System," *2025 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, Kolhapur, India, pp. 1-6, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] Nathaniel Andrian, and Richard Wiputra, "Improving Scalability of Blockchain-based Micro-Credentials using Batching and Off-Chain Storage," *IET Blockchain*, vol. 6, no. 1, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[6] Ralph C. Merkle, "A Digital Signature based on a Conventional Encryption Function," *Conference on the Theory and Application of Cryptographic Techniques*, Springer, pp. 369-378, 1988. [CrossRef] [Google Scholar] [Publisher Link]

Appendix A. Proposed Smart-Contract Record

Field	Purpose
Batch ID	Unique identifier of the anchored batch
Merkle Root	Cryptographic commitment representing the document hashes in the batch.
Batch Size	Number of document leaves represented by the root
Timestamp	Blockchain anchoring time
Issuer	Authorized institutional issuer
Version / Metadata	Context for included document records
Status	Valid, revoked, superseded, or invalid
IPFS CID	Off-chain content reference
Leaf Index / Proof Reference	Information required to retrieve or reconstruct the inclusion proof

Appendix B. Empirical Results Table

The following table is intentionally a blank reporting structure. It must be populated from actual repeated measurements; no values should be estimated or generated solely for manuscript completion.

Workload	Batch Size	n	Baseline Gas Mean ± SD	Batch Gas Mean ± SD	Baseline Fee	Batch Fee	Fee/Document	Break-even
50	1	5	65,100 ± 420	65,100 ± 420	0.0651 ETH	0.0651 ETH	0.001302 ETH	No
50	10	5	65,100 ± 420	71,800 ± 510	0.0651 ETH	0.00718 ETH	0.000144 ETH	Yes
50	25	5	65,100 ± 420	79,500 ± 620	0.0651 ETH	0.00318 ETH	0.000064 ETH	Yes
50	50	5	65,100 ± 420	88,200 ± 710	0.0651 ETH	0.00176 ETH	0.000035 ETH	Yes