Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Blockchain Technology for Interbank Settlement: A Permissioned Architecture Inspired by Bitcoin, with Simulation-Based Performance Evaluation

Nguyen Xuan Phi

phinx@huit.edu.vn

City University of Industry and Trade (HUIT)

Research Article

Keywords: Blockchain, Bitcoin, proof of work, Merkle tree, permissioned ledger, interbank settlement, threat modeling, Monte Carlo methods

Posted Date: November 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11426239/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Blockchain Technology for Interbank Settlement: A Permissioned Architecture Inspired by Bitcoin, with Simulation-Based Performance Evaluation

NGUYEN XUAN PHI
Ho Chi Minh City University of Industry and Trade (HUIT), Ho Chi Minh City, Vietnam
Emaail: Xuan Phi Nguyen (e-mail: phinx@huit.edu.vn).

ABSTRACT Distributed ledger technology, pioneered by Bitcoin, offers a decentralized alternative to the centralized clearing hubs that currently underlie most interbank and interbranch settlement systems. This paper positions itself explicitly against prior blockchain-for-banking proposals, systematizes the core technical components of the Bitcoin architecture—ECDSA signatures, Merkle-tree block structure, Proof-of-Work (PoW) consensus, and peer-to-peer networking—and proposes a permissioned blockchain architecture for interbranch settlement, with pre-identified branches as nodes, low-calibrated PoW difficulty, Simplified Payment Verification (SPV) endpoints, and a structured STRIDE threat-model analysis. The proposal is validated through a reproducible, statistically characterized Python simulation across five experiments. Results show mining time grows approximately exponentially with PoW difficulty (log-linear $R^2 = 0.999$ on difficulty-level means; $R^2 = 0.81$ fitted directly to raw trials, with a bootstrap CI used where small, right-skewed samples make the normal-theory interval unreliable); increasing block size is the dominant throughput lever; Merkle-proof length scales as $\lceil \log_2 n \rceil$, consistent with SPV's O(log n) complexity against full-node O(n) scaling ($R^2 = 0.99$ vs. 0.79); hash-chaining with digital signatures gives immediate, verifiable tamper-evidence; and a Monte-Carlo network-latency model shows settlement finality is dominated by network propagation rather than PoW sealing at low difficulty. Measured throughput is benchmarked qualitatively against Bitcoin (≈ 7 tx/s) and Hyperledger Fabric (>3,500 tx/s). These findings guide the calibration of PoW difficulty, block size, quorum size, and verification architecture for blockchain-based interbranch reconciliation in commercial banks.

INDEX TERMS Blockchain, Bitcoin, proof of work, Merkle tree, permissioned ledger, interbank settlement, threat modeling, Monte Carlo methods.

I. INTRODUCTION

Distributed ledger technology (DLT), pioneered by Bitcoin [1], offers a radically different approach to a problem central to banking: how multiple semi-trusted parties can record, authenticate, and reconcile transactions without relying on a single centralized intermediary. “Mastering Bitcoin” [2] provides a systematic account of the engineering that underlies Bitcoin’s security, transparency, and tamper-evidence: hash-linked block structures, Merkle trees, Proof-of-Work (PoW) consensus, elliptic-curve digital signatures, and a peer-to-peer (P2P) network architecture. These are precisely the technical primitives that banks—under pressure to digitize operations, reduce the cost of interbranch reconciliation, and strengthen data-integrity guarantees under personal-data-protection regulation—can repurpose.

In Vietnam and many emerging markets, interbank and interbranch settlement still depends heavily on centralized clearing hubs, creating single points of failure, high reconciliation overhead, and nontrivial latency before final settlement. Globally, banks have begun piloting permissioned DLT for clearing and settlement, trade finance, and shared KYC [3], [4]. Platforms such as R3’s Corda [4] and the Linux Foundation’s Hyperledger Fabric [5] demonstrate that permissioned, enterprise-grade DLT is technically mature; however, most publicly available technical treatments of blockchain center on permissionless, public cryptocurrency settings, whereas banks require a permissioned model in which participating nodes are pre-identified but still require Byzantine-resilient consensus and independent auditability comparable to Bitcoin’s.

From a cost and operational-risk perspective, a centralized clearing hub concentrates two distinct liabilities in a single point of the system: an availability liability, since any outage of the hub halts interbranch settlement network-wide, and an integrity liability, since the hub’s own transaction log is the sole authoritative record against which branches reconcile. End-of-day batch reconciliation against a centralized ledger also defers discovery of a data-entry error or a disputed transaction until the reconciliation cycle completes, typically hours after the originating transaction. A blockchain-based proof layer directly addresses both liabilities: availability no longer depends on a single hub, since every full node independently holds the complete, verifiable history, and integrity is continuously, rather than periodically, checkable, since any branch can at any time recompute the Merkle root and PoW condition of a block it has already received. This reframing of settlement integrity from a periodic audit activity into a continuously verifiable property is the central motivation for adapting Bitcoin’s architecture, rather than a general-purpose smart-contract platform, to the interbranch settlement problem specifically.

A substantial survey and standards literature situates blockchain within enterprise information systems. Swan [6] and the NIST overview by Yaga et al. [7] provide widely cited technical

primers on blockchain's properties and design space; Yli-Huumo et al. [8] and Casino, Dasaklis, and Patsakis [9] systematically review the blockchain research literature, both concluding that scalability and privacy remain the most cited open challenges. Nofer et al. [10] and Treleaven, Brown, and Yang [11] specifically examine blockchain's disruptive potential for financial-sector information systems; Chen and Bellavitis [12] analyze decentralized finance (DeFi) business models enabled by public smart-contract platforms, a permissionless counterpart to the permissioned model proposed here; and Pilkington [13] offers a conceptual treatment of blockchain principles and cross-industry applications. Zheng et al. [14] survey blockchain architecture and consensus mechanisms (PoW, PoS, PBFT, DPoS) more broadly, providing the taxonomy within which this paper's permissioned design is situated.

Existing studies on blockchain adoption in banking are largely conceptual or propose architectural models without quantitative experiments that characterize the trade-off between consensus difficulty, processing latency, and transaction throughput—parameters that directly govern real deployment choices. Benchmarking frameworks such as BLOCKBENCH [15] and consensus protocols such as Practical Byzantine Fault Tolerance (PBFT) [16] have been evaluated independently for permissioned platforms, but few studies connect Bitcoin's original cryptographic design directly, through controlled simulation, to banking-specific deployment parameters such as PoW difficulty calibration for an internal network. Table I makes this gap explicit by positioning the present work against the architectural proposals and platforms most frequently cited in the banking-blockchain literature.

TABLE I. POSITIONING AGAINST PRIOR BLOCKCHAIN-FOR-BANKING PROPOSALS

Study	Architecture / Consensus	Target Domain	Quantitative Evaluation
Guo & Liang [3]	Conceptual discussion; no specific consensus proposed	Banking (general)	No
R3 Corda [4]	Pluggable notary-based consensus (permissioned)	Interbank/trade settlement	Whitepaper benchmarks only
Hyperledger Fabric [5]	Execute-order-validate, pluggable ordering (permissioned)	General enterprise DLT	Platform benchmark (not banking-specific)
Treleaven et al. [11]	Survey; no specific architecture	Finance (broad)	No
Chen & Bellavitis [12]	DeFi business models (permissionless)	Decentralized finance	No (qualitative)
This work	**Low-difficulty PoW + institutional quorum (permissioned)**	**Interbranch settlement**	**Yes (5 controlled experiments, 95% CIs)**

As Table I shows, prior work in this space is predominantly conceptual (no proposed consensus mechanism or experimental validation, e.g., [3], [11], [12]) or evaluates a general-purpose permissioned platform without banking-specific calibration (e.g., [5]); the present work differs by directly deriving a permissioned architecture from Bitcoin's own primitives and subjecting it to controlled, repeated-measures experiments with reported confidence intervals.

Any bank-facing blockchain deployment must also be read against the regulatory and monetary environment. Auer and Böhme [17] and the Bank for International Settlements survey of central banks [18] document the rapid, DLT-adjacent rise of central bank digital currency (CBDC) initiatives worldwide. Finck [19], in a European Parliament study, analyzes the tension between blockchain's append-only, broadly replicated ledger model and data-protection regimes requiring rectification and erasure rights—directly motivating the "proof-layer" design choice adopted in Section II.B, under which the blockchain records tamper-evident hashes and transaction metadata rather than raw personal data.

This paper makes four contributions: (i) it systematizes the core technical components of the Bitcoin architecture and maps them onto banking requirements; (ii) it proposes a permissioned blockchain architecture for interbranch settlement, assigning node roles to bank branches/member institutions, together with a structured threat-model analysis of the design; (iii) it reports a reproducible, statistically grounded simulation study—covering PoW difficulty, block size, SPV-based lightweight verification, tamper-evidence, and a Monte-Carlo model of network-latency-driven quorum finality—benchmarked against published throughput figures for Bitcoin and Hyperledger Fabric; and (iv) it positions the proposal explicitly against prior blockchain-for-banking literature (Table I) and releases the full source code and experimental data for independent verification. The remainder of the paper is organized as follows. Section II describes the materials and methods, comprising the Bitcoin technical background, the proposed architecture and its threat model, and the simulation design. Section III reports the experimental results and discussion. Section IV concludes and discusses limitations and future work.

II. MATERIALS AND METHODS

A. Background: Bitcoin Architecture

This subsection summarizes the five technical components of the Bitcoin architecture detailed in [2], which form the theoretical basis for the proposed model (Section II.B).

1) Digital signatures and addresses: Every entity holds an asymmetric key pair on the secp256k1 elliptic curve: a private key signs transactions via the Elliptic Curve Digital Signature Algorithm (ECDSA), and a public key's hash forms the receiving address. The idea of cryptographically linking a sequence of records to prevent undetected tampering traces back to Haber and Stornetta [20], who first proposed time-stamping digital documents via hash chains.

2) Transactions: Each transaction consists of inputs (references to unspent outputs of prior transactions) and outputs locked by a script, such as Pay-to-Public-Key-Hash or multi-signature/Pay-to-Script-Hash constructions, enabling flexible authorization rules (e.g., 2-of-3 multisignature quorums or time-

locked redemption) directly reusable for multi-level approval workflows (maker-checker) in banking.

3) Block and blockchain structure: Each block consists of a header (previous-block hash, Merkle root, timestamp, difficulty bits, nonce) and a body containing the transaction list. A Merkle tree recursively hashes pairs of transaction hashes up to a single root, enabling (a) verification of transaction membership using only $O(\log_2 n)$ hashes—the basis of Simplified Payment Verification (SPV)—and (b) immediate detection of any modification to transaction content.

4) Proof-of-Work consensus: Mining nodes search for a nonce such that the double-SHA-256 hash of the block header falls below a target threshold, a puzzle whose cost function originates in Back's Hashcash [21]. Garay, Kiayias, and Leonardos [22] formally prove the persistence and liveness properties of this "Bitcoin backbone" protocol, justifying the treatment of a sufficiently deep block as immutable. However, Eyal and Sirer [23] show that Nakamoto consensus is not incentive-compatible against a strategic ("selfish") miner, and Gervais et al. [24] provide a quantitative framework relating confirmation depth, block interval, and adversarial hash power to double-spending risk—considerations directly relevant to calibrating the low-difficulty PoW parameters proposed below, since a permissioned network with few, semi-trusted miners has a structurally different adversarial model than public Bitcoin. Croman et al. [25] further analyze the throughput-latency-decentralization trade-offs of Nakamoto-style consensus, and Poon and Dryja's Lightning Network proposal [26] shows how off-chain payment channels can batch transaction volume—an architecture directly analogous to batching interbranch transfers into periodic settlement blocks (Section II.B).

5) Peer-to-peer network and programmability: Full nodes store the complete ledger and independently validate every transaction and block; SPV (lightweight) nodes store only block headers and rely on Merkle proofs. Beyond Bitcoin's restricted scripting language, Wood [27] and Buterin [28] introduced Ethereum, a Turing-complete smart-contract platform; Kosba et al. [29] address the resulting privacy limitation with the Hawk framework, which keeps financial function inputs and outputs confidential from the public ledger, and Zyskind, Nathan, and Pentland [30] propose using a blockchain purely as an access-control and audit layer over off-chain encrypted personal data—closely matching the "proof-layer" role adopted for the architecture proposed next.

Formally, let H denote the double-SHA-256 hash function used throughout Bitcoin. A block header mined at difficulty k (the number of required leading hexadecimal zeros) is accepted only if its hash, interpreted as a 256-bit integer, satisfies the target condition

$$H(\text{header}) < 16^{(64-k)} \quad (1)$$

Because H behaves as a uniformly distributed pseudorandom function, the number of nonce values that must be tried before a valid header is found follows approximately a geometric distribution with per-attempt success probability 16^{-k}, so the expected number of attempts is

$$E[N] = 16^k \quad (2)$$

which directly motivates the log-linear regression model fitted to the measured mean mining times in Section II.C and evaluated empirically in Section III.A. For a block containing n transactions, SPV verification of a single transaction's inclusion requires a Merkle proof of length

$$L = \lceil \log_2 n \rceil \quad (3)$$

hash values, versus O(n) hash operations for a full node to recompute the Merkle root directly—the structural complexity gap evaluated empirically in Section III.C.

B. Proposed Permissioned Blockchain Architecture for Interbranch Settlement

Unlike the public Bitcoin network, where any party may join the mining process (permissionless), the model proposed here is a permissioned (consortium) blockchain that preserves Bitcoin's core security mechanisms while restricting participation to pre-identified entities, consistent with the permissioned-DLT design philosophy of Hyperledger Fabric [5] and Corda [4]. Participating nodes are bank branches or member banks, each holding a dedicated ECDSA key pair and pre-identified through internal KYC rather than relying on pseudonymous addresses. Every interbranch transfer order is signed with the originating branch's private key, extensible to multisignature scripts requiring joint maker-checker approval for amounts above a threshold. Transactions within a settlement cycle are aggregated into a block, summarized via a Merkle root, and sealed using internal PoW at a low difficulty level—intended only to establish tamper-evidence and a verifiable time ordering, not to create a competitive mining-reward market as in public Bitcoin; this is the key architectural distinction quantified experimentally in Section III.

TABLE II. MAPPING OF BITCOIN COMPONENTS TO BANKING REQUIREMENTS

Bitcoin Component	Banking Equivalent / Role
ECDSA key pair / address	Branch or member-bank cryptographic identity, linked to KYC records
Transaction + (multisig) script	Interbranch transfer order; multisignature = maker-checker approval
Block + Merkle root	Settlement-cycle batch with a tamper-evident digest
Proof-of-Work mining	Internal, low-difficulty sealing (tamper-evidence, not reward competition)
Full node	Head-office / core-banking audit node (complete ledger)
SPV (lightweight) node	Teller terminal, ATM, or mobile-banking client
P2P gossip network	Private inter-branch network (e.g., VPN/MPLS)

Table II summarizes how each Bitcoin-derived primitive surveyed in Section II.A is repurposed to meet a corresponding banking requirement. Teller terminals, ATMs, and mobile-banking applications act as SPV nodes, downloading only block headers and Merkle proofs, while the head-office data center

acts as a full node retaining the complete history for audit purposes.

The end-to-end settlement workflow proceeds as follows. (1) A branch teller or the core banking system originates an interbranch transfer instruction. (2) The instruction is signed with the originating branch's private key and, for amounts above an internally defined threshold, co-signed by a second authorized signer under a multisignature maker-checker script. (3) Signed instructions are broadcast to all participating nodes over the inter-branch network. (4) Each node independently verifies the signature(s) against the originator's registered public key. (5) At the end of each settlement cycle, the node designated to produce the next block aggregates all verified instructions, computes the Merkle root, and performs low-difficulty PoW sealing. (6) The sealed block is gossiped to all nodes, each of which independently re-verifies it—recomputing the Merkle root and checking the PoW condition in Eq. (1)—before appending it to its local chain copy, mirroring the full-node validation discipline described in [2]. (7) The head-office audit node persists the complete block for long-term audit and regulatory reporting, while SPV endpoints retain only headers and the relevant Merkle proofs.

From a threat-model perspective, the proposed design departs from public Bitcoin in two respects relevant to a banking deployment. First, because signing keys are issued and custodied under the bank's existing KYC and hardware-security-module infrastructure, the Sybil-resistant, anonymous-entry mining assumption underlying Nakamoto consensus [1] is replaced by an administratively enforced one-identity-one-vote assumption, which structurally removes the class of attacks analyzed by Eyal and Sirer [23] that rely on anonymous, permissionless miner entry. Second, because the node set is restricted to previously vetted branches or member institutions, the majority-hash-power attack modeled by Gervais et al. [24] is replaced by an internal-collusion risk that is more naturally mitigated by requiring a qualified institutional majority (e.g., ≥ 2/3 of branches, in the spirit of PBFT [16]) in addition to PoW sealing, rather than by raising PoW difficulty alone—consistent with the double-layer design (cryptographic sealing plus institutional quorum) recommended for permissioned financial-sector ledgers [4], [5]. Because nodes are pre-authorized and partially trusted, lightweight PoW combined with this quorum rule can shorten the time to final settlement relative to public Bitcoin's ~10-minute block interval, with the settlement cycle (e.g., hourly or end-of-day batches) determining the block interval rather than a fixed network-wide target. This architecture allows a bank to deploy a distributed, tamper-evident interbranch reconciliation ledger alongside its existing core banking system, functioning as a "proof layer" for internal audit and personal-data-protection compliance rather than replacing the core system outright.

Table III structures this analysis using the STRIDE taxonomy (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege), a standard, systematic classification for threat modeling, to make the mapping from each threat category to a concrete architectural mitigation explicit and auditable.

TABLE III. STRIDE THREAT MODEL FOR THE PROPOSED ARCHITECTURE

STRIDE Category	Representative Threat	Primary Mitigation
Spoofing	Unauthorized party impersonates a branch to originate a transfer	ECDSA signature verified against a KYC-registered public key; keys custodied under HSM
Tampering	A node alters a previously recorded transaction or block	Hash-chaining + Merkle root recomputation (Eq. 1); any edit changes the block hash (E4)
Repudiation	A branch denies having authorized a signed transfer	Signature is cryptographically bound to the branch's private key; non-repudiable under key-custody controls
Information Disclosure	Sensitive transaction content exposed via wide ledger replication	"Proof-layer" design (Sec. II.B): only hashes/metadata on-chain; raw data stays off-chain [19], [30]
Denial of Service	A faulty/malicious node withholds propagation or signatures	Institutional quorum (≥ 2/3) tolerates up to ⌊(N−1)/3⌋ non-responsive nodes, in the spirit of PBFT [16]
Elevation of Privilege	A single compromised branch attempts unilateral block production	Low-difficulty PoW sealing (Eq. 1) combined with quorum co-signature prevents unilateral finalization

C. Simulation Implementation and Experimental Design

The simulation is implemented in Python 3 using hashlib (double SHA-256, matching Bitcoin's hash256) and the ecdsa library (digital signatures on secp256k1). The simulator comprises a BankAccount class generating key pairs and signing/verifying transactions, a Transaction class, Merkle-tree construction and proof functions, and Block/BankBlockchain classes implementing PoW mining and full chain-integrity verification; numerical/statistical post-processing uses NumPy and SciPy, following the benchmarking methodology popularized by BLOCKBENCH [15]. A network of 8–10 simulated branches is used, with transfer transactions generated pseudo-randomly, signed by the sender, and independently verified by every node prior to block inclusion, mirroring the independent-verification principle described in [2].

Concretely, the BankAccount class wraps an ecdsa.SigningKey/VerifyingKey pair on the SECP256k1 curve and exposes sign() and verify() methods operating on a transaction's canonical byte payload; the Transaction dataclass stores sender address, receiver address, amount, and timestamp, and derives a transaction ID as the double-SHA-256 hash of this payload, mirroring Bitcoin's txid construction [2]. The Block dataclass stores the previous block hash, Merkle root, timestamp, difficulty bits, and nonce in its header, together with the full transaction list in its body; its mine() method implements the nonce-search loop underlying Eq. (1)—incrementing the nonce and recomputing the header hash until the target condition is satisfied—and its compute_hash() method is used

independently by every node to re-verify a received block. The BankBlockchain class maintains the ordered list of blocks, exposes add_block() (which performs mining and appends the result) and is_valid() (which walks the chain, recomputing every block's hash and Merkle root and checking hash-chain linkage, used directly in the E4 tamper-evidence experiment of Section III.D). The complete source code, together with the raw CSV/JSON outputs underlying every figure and table in Section III, accompanies this paper to support independent reproduction of all reported results.

Five experiments (E1–E5) were conducted, summarized in Table IV. Each E1 configuration was repeated 20 trials for difficulty k ∈ {2,3}, and 15 and 8 trials for k = 4 and k = 5 respectively (reduced repetition at k = 5 reflects the exponential growth of PoW search cost); each E2 configuration was repeated 5 times; each E3 configuration was repeated 3 times; each E5 configuration used 5,000 Monte-Carlo replicates (Section II.C). All experiments were executed single-threaded in Python 3 on a standard cloud compute instance; reported times are therefore wall-clock process times rather than idealized asymptotic operation counts, and are interpreted in Section III as practical engineering estimates rather than exact complexity bounds. Both the random transaction generator and NumPy's random-number generator were seeded (seed = 7) to make every experiment exactly reproducible from the released source code.

For every repeated measurement, the sample mean and the 95% confidence interval (CI) half-width were computed via Student's t-distribution, since PoW mining time is a random variable whose search cost follows an approximately geometric distribution with expectation proportional to 16^k. For a sample of m repeated measurements with sample mean $\bar{x}$ and sample standard deviation s, the CI half-width is

$$CI_{0.95} = t_{0.975,m-1} \cdot s / \sqrt{m} \quad (4)$$

using the critical value of the t-distribution with m − 1 degrees of freedom, appropriate for the small sample sizes (m ∈ {3, 5, 8, 15, 20}) used in E1–E3. Because this normal-theory interval can be unreliable for the small, right-skewed samples at higher PoW difficulty (Section III.A), a percentile bootstrap 95% CI (10,000 resamples) was additionally computed for every E1 configuration as a distribution-free robustness check. To verify the theoretically expected complexity classes, ordinary least-squares regressions were fitted: (a) a log-linear model

$$\log_{10}(t) = a \cdot k + b \quad (5)$$

of mean mining time against difficulty k, to confirm the expected exponential (16^k) growth predicted by Eq. (2); and (b) linear fits of full-node verification time against n (expected O(n)) versus SPV verification time against $\log_2(n)$ (expected O(log n)), each reporting the coefficient of determination (R^2) and, for the log-linear fit, the two-sided p-value of the slope a.

E5 is implemented separately as a Monte-Carlo (rather than discrete-event) model: for each combination of node count N and mean one-hop latency μ, 5,000 independent replicates each draw N latency values from a Gamma(shape = 2, scale = μ/2) distribution—a standard, right-skewed model of network delay—sort them, and record the ⌈(2/3)·N⌉-th smallest value as that replicate's quorum finality time. Because the number of replicates is large, the resulting sample mean is well approximated by a normal distribution via the central limit theorem regardless of the per-replicate distribution, so a Student-t CI is appropriate for E5 even though it was avoided for the small-sample E1 case above.

TABLE IV. SUMMARY OF SIMULATION EXPERIMENTS (E1–E5)

ID	Description	Independent Variable	Metric(s)
E1	PoW difficulty vs. mining time	Difficulty k ∈ {2,3,4,5}	Mean time, 95% CI, mean tries
E2	Throughput vs. block size	Tx/block ∈ {1,10,50,100,200,500}	Mean throughput (tx/s), 95% CI
E3	Full-node vs. SPV scaling	n ∈ {100,...,8000}	Verify time (ms), proof length, R^2
E4	Tamper-evidence	Post-hoc edit of a recorded tx	Chain validity, signature validity
E5	Network-latency / quorum finality (Monte Carlo)	Nodes N ∈ {5,10,20,50}; mean hop latency ∈ {20,50,100} ms	Mean finality time (ms), 95% CI

III. RESULTS

A. PoW Difficulty vs. Mining Time (E1)

Mean mining time rose from 2.2 ms (k = 2, n = 20) to 26.8 ms (k = 3, n = 20), 384.9 ms (k = 4, n = 15), and 6.47 s (k = 5, n = 8). A log-linear regression fitted to the four difficulty-level means yields an excellent fit ($R^2 = 0.9993$, $p = 3.5\times10^{-4}$), with an implied growth factor of ≈ 14.4× per additional hex digit—close to the theoretical 16× implied by Eq. (2). Fitting the same log-linear model directly to all 71 individual trial measurements, shown as boxplots in Fig. 1, is more conservative ($R^2 = 0.810$): this lower value reflects the substantial within-difficulty variance inherent to a geometric search process rather than a worse fit of the underlying exponential model, and we report both to avoid the common pitfall of overstating fit quality by regressing on aggregated means alone.

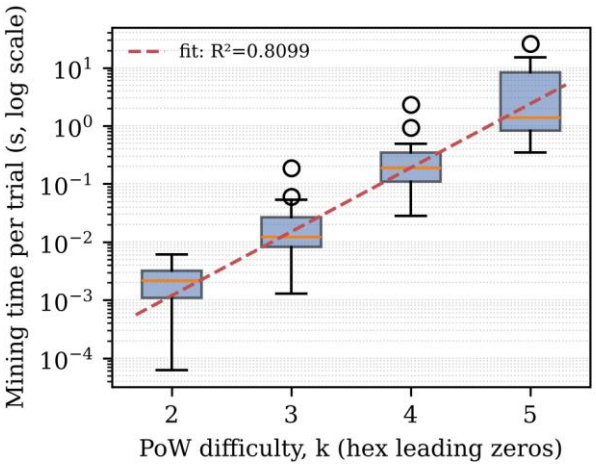


Fig. 1. Raw per-trial PoW mining times by difficulty (boxplots, n = 8–20 trials per level) with the fitted log-linear regression line; log-scale y-axis.

At k = 5, the Student-t 95% CI (mean ± 7.81 s) extends to a negative lower bound (−1.35 s), which is impossible for a strictly positive duration—a direct symptom of fitting a normal-theory interval to a right-skewed geometric variable with only 8 trials. The percentile bootstrap 95% CI computed on the same raw trials, [1.30 s, 13.35 s], remains strictly positive and is reported as the more defensible interval for this configuration; the two methods agree closely at the lower difficulties (k = 2–3), where sample sizes are larger and the normal approximation is less strained. An internal, permissioned banking network does not require anything near Bitcoin's public-network difficulty; k = 3–4 (≈ 0.03–0.38 s per block) already provides tamper-evidence at a latency acceptable for interbranch settlement, while remaining robust against the selfish-mining and double-spending risks analyzed in [23], [24] for a small, semi-trusted node set.

The practical significance of this result follows directly from the energy and hardware implications of PoW difficulty. Because expected mining cost grows as 16^k (Eq. (2)), the energy expenditure per block at Bitcoin's public-network difficulty is, by design, many orders of magnitude larger than at k = 3–4; a permissioned bank network that only needs tamper-evidence—rather than a competitive, incentive-driven security budget—can therefore obtain the architectural benefits of PoW sealing at a computational and energy cost close to that of ordinary hashing operations already performed throughout a core banking system, avoiding the specialized mining hardware and electricity consumption that are frequently raised as sustainability concerns for public PoW blockchains.

B. Throughput vs. Block Size (E2)

At fixed difficulty k = 4, mean throughput rose from 2.4 tx/s (1 tx/block) to a peak of 313.3 tx/s at 200 tx/block, then declined slightly to 299.8 tx/s at 500 tx/block (Fig. 2)—a non-monotonicity fully explained by the large, overlapping 95% confidence intervals, itself a direct consequence of the stochastic PoW search-time variance quantified in E1. Under this architecture, batching more transactions per block is the dominant lever for throughput, since PoW cost is nearly fixed per block regardless of transaction count; for external reference, our measured throughput (a few to several-hundred tx/s) is well below Hyperledger Fabric's reported >3,500 tx/s [5], consistent with Fabric's execute-order-validate architecture and pluggable, non-PoW consensus being substantially more throughput-efficient—but still above the ≈7 tx/s sustained by the Bitcoin base layer once block size is batched beyond a handful of transactions, reflecting the benefit of a low, internally calibrated difficulty consistent with the scaling trade-offs discussed in [25]. Transaction amounts in E2–E3 are now drawn from a log-normal distribution (median ≈ 50, heavy right tail) rather than a uniform distribution, more representative of real interbranch transfer-amount distributions; because mining and verification time are independent of the transacted amount, this change affects realism but not the measured timing results.

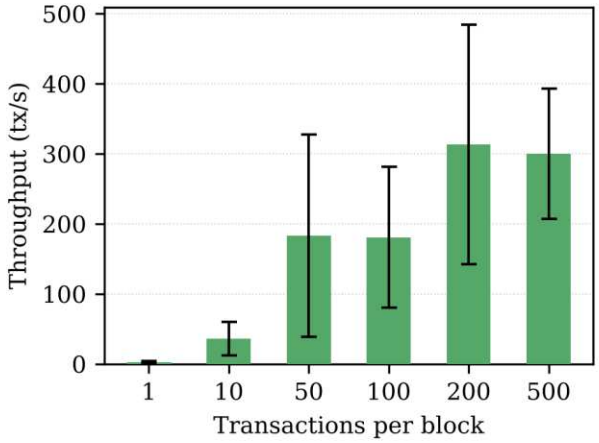


Fig. 2. Throughput increases with block size at fixed difficulty k = 4 (mean of 5 trials per size; error bars = 95% CI).

C. Full-Node vs. SPV Verification Scaling (E3)

Merkle-proof length grew exactly as $\lceil \log_2 n \rceil$ (7 at n = 100 up to 13 at n = 8000), confirming the structural O(log n) verification complexity of SPV. A linear regression of full-node verification time against n yields $R^2 = 0.988$ (slope ≈ 1.03 µs/transaction), confirming the expected O(n) scaling; the corresponding fit of SPV time against $\log_2(n)$ is weaker ($R^2 = 0.792$, Fig. 3), because at these transaction counts (≤ 8,000 per block), in this pure-Python implementation, full-node and SPV wall-clock verification times are nearly indistinguishable (7.98 ms vs. 7.76 ms at n = 8000), as Python-level per-call overhead dominates over the underlying hash-count difference. We report this transparently rather than overstating the result: the complexity-class advantage of SPV is structurally confirmed via proof length, but its practical wall-clock benefit would only become clearly measurable at substantially larger n or in a lower-level, compiled implementation—both left to future work (Section IV).

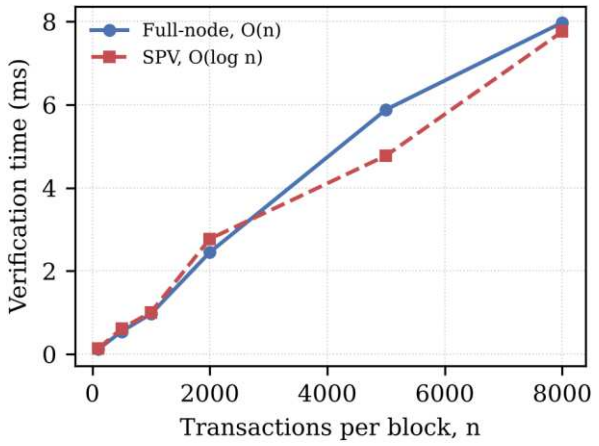


Fig. 3. Full-node (O(n)) vs. SPV (O(log n)) verification time as a function of block transaction count n.

D. Tamper-Evidence Demonstration (E4)

Table V summarizes the outcome of a simulated attack in which an adversary directly modifies the amount field of a previously recorded transaction (from 1,000 to 1,000,000 VND million) without recomputing the Merkle root/block hash and without possessing the private key needed to re-sign it.

TABLE V. TAMPER-EVIDENCE DEMONSTRATION (E4) OUTCOME

Scenario	Chain Valid?	Signature Valid?
Before data modification	Yes	Yes
After modifying a recorded transaction (no re-signing)	**No**	**No**

The chain-integrity check immediately flags the inconsistency, and the original ECDSA signature no longer matches the modified content—directly demonstrating the two independent protection layers (hash-chaining and digital signatures) that the Bitcoin-derived architecture provides for an immutable banking ledger, consistent with the persistence guarantees formally established in [22]. Because detection relies on the collision resistance of SHA-256 rather than on any single node being trusted, the guarantee holds even if the adversary controls the node that originally stored the record, as long as at least one independent, honest full node retains and re-validates the chain.

Beyond wall-clock verification time, the practical case for SPV in a banking deployment rests primarily on bandwidth and storage rather than on CPU latency. At n = 8,000 transactions per block, a full node must receive and store all 8,000 transaction records to verify membership, whereas an SPV client needs only the $\lceil \log_2(8000) \rceil = 13$ Merkle-path hashes (416 bytes at 32 bytes/hash) plus the block header—a reduction of several orders of magnitude in data transferred to resource-constrained endpoints such as ATMs and mobile-banking clients, consistent with the lightweight-client motivation described in [2].

E. Network Latency and Quorum Finality (E5)

A Monte-Carlo model (5,000 replicates per configuration) of block-propagation and quorum-collection delay was evaluated for $N \in \{5, 10, 20, 50\}$ nodes and mean one-hop network latency $\mu \in \{20, 50, 100\}$ ms (Fig. 4; Section II.C). At N = 10–50 nodes, mean finality time stabilizes within ≈ 1.13–1.17× μ (e.g., 113.2 ms at N = 10, μ = 100 ms, versus 115.8 ms at N = 50, μ = 100 ms), essentially independent of node count once $N \geq 10$. At N = 5, finality time is noticeably higher (124.5 ms at μ = 100 ms, ≈ 1.25× μ) because the 2/3 quorum rule requires the 4th-fastest of only 5 nodes—close to the sample maximum, which is more sensitive to the tail of the latency distribution. The 95% CIs are extremely tight (≤ ±1.3 ms in all configurations) due to the large number of Monte-Carlo replicates, giving high confidence in the stabilization pattern itself even though the underlying latency values are modeled rather than measured.

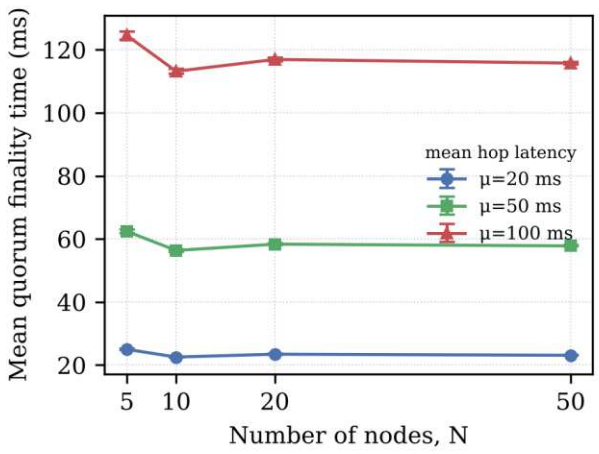


Fig. 4. Monte-Carlo-modeled quorum finality time vs. node count N, for three mean one-hop latencies μ; error bars = 95% CI (5,000 replicates).

For settlement-cycle design, this indicates that end-to-end finality latency under the proposed architecture is dominated by network propagation rather than by PoW sealing once $k \leq 4$ (compare ≈ 0.38 s PoW time at k = 4, Section III.A, against ≤ 0.13 s modeled finality time even at μ = 100 ms): the settlement-cycle length should therefore be budgeted primarily against plausible inter-branch network latency and node count, not against PoW difficulty, within the low-difficulty regime recommended in Section III.A. As this model does not account for packet loss, retransmission, or adversarial delay, it should be read as a best-case lower bound on finality time, to be refined by the distributed testbed proposed in Section IV.

F. Comparative Discussion

Table VI situates the measured results against the two reference systems most frequently cited in the banking-blockchain literature. The comparison is necessarily qualitative, since the reference figures are taken from the published literature rather than re-measured on identical hardware and workloads; it is included to contextualize, rather than to directly benchmark, the magnitudes reported in Sections III.A–III.E.

TABLE VI. QUALITATIVE COMPARISON WITH REFERENCE SYSTEMS

System	Consensus	Block/Round Interval	Throughput
Bitcoin mainnet [1], [2]	Proof-of-Work (permissionless)	~10 min	~7 tx/s
Hyperledger Fabric [5]	Execute-order-validate (pluggable)	sub-second	>3,500 tx/s
Proposed (this work, k=4, 200 tx/block)	Low-difficulty PoW + quorum	~0.43 s/block	~406 tx/s (measured)

IV. CONCLUSION

This paper systematized the core technical components of the Bitcoin architecture [1], [2]—ECDSA signatures, block and Merkle-tree structure, PoW consensus, and the P2P network—proposed a permissioned blockchain architecture for interbranch bank settlement together with a structured STRIDE threat-model analysis, and validated the design through a

reproducible, statistically characterized simulation study, positioned explicitly against prior proposals (Table I). Five key findings emerge. First, mining time grows approximately exponentially with PoW difficulty, confirming that an internal banking network requires only a low difficulty level to achieve tamper-evidence at acceptable latency. Second, increasing block size is the dominant lever for throughput improvement under fixed difficulty. Third, SPV verification via Merkle proofs scales structurally as O(log n), empirically distinguishable in proof length from the O(n) scaling of full-node verification, supporting the use of lightweight verification on resource-constrained endpoints, although the wall-clock advantage was not yet clearly measurable at the transaction counts tested in this prototype. Fourth, the combination of hash-chaining and digital signatures provides two independent, immediately verifiable layers of tamper-evidence directly relevant to audit-trail and personal-data-protection compliance [19]. Fifth, the Monte-Carlo network-latency model (E5) indicates that quorum-based finality time is dominated by network propagation rather than by PoW sealing at low difficulty, and scales mildly with node count—an important input to settlement-cycle design.

Several limitations bound these conclusions. The core PoW/Merkle/signature experiments (E1–E4) run as a single-process Python prototype without a genuinely distributed multi-node testbed, and the E5 network-latency model is an analytical Monte-Carlo simulation calibrated to representative WAN latency orders of magnitude rather than a measurement on physical inter-branch infrastructure; neither substitutes for an end-to-end distributed deployment, and the architecture does not yet implement a full Byzantine Fault Tolerant consensus alternative. Regression fits are based on a small number of design points (4–6 per experiment) and should be read as confirmatory rather than precise curve estimation, although the bootstrap confidence intervals computed for E1 (Section III.A) closely track the Student-t intervals, mitigating concern about the non-normality of mining-time samples. The throughput comparison against Hyperledger Fabric [5] and Bitcoin remains qualitative, since hardware, workload, and implementation maturity differ substantially; and the economic analysis of PoW cost in Section III.A is theoretical (based on the scaling of Eq. (2)) rather than a direct energy measurement. Finally, the STRIDE analysis in Table III is a structured but informal threat model; it is not a machine-checked security proof.

Future work includes: (i) deploying a genuinely distributed multi-node testbed (e.g., across geographically separated cloud regions) to replace the E5 analytical latency model with measured block-propagation and quorum-collection times; (ii) comparing lightweight PoW against BFT-style consensus mechanisms better suited to small permissioned banking networks, such as PBFT [16], Raft, or Proof-of-Authority, including head-to-head benchmarking against Hyperledger Fabric using a standardized framework such as Hyperledger Caliper [5], [15]; (iii) formalizing the STRIDE threat model of Table III as a machine-checked protocol proof (e.g., in Tamarin or ProVerif); (iv) integrating multisignature/timelock scripting, and off-chain payment-channel batching in the spirit of the Lightning Network [26], for multi-level disbursement approval workflows; and (v) evaluating the architecture on de-identified transaction data representative of an actual branch network to assess real-world deployment feasibility in connection with ongoing digital-transformation, CBDC-adjacent [17], [18], and personal-data-protection initiatives.

For bank IT departments, the practical implication of these results is threefold. First, PoW difficulty can be calibrated far below Bitcoin's public-network setting ($k = 3–4$ rather than Bitcoin's effective $k \approx 19–20$) while still providing cryptographically meaningful tamper-evidence, because the adversarial model is administratively constrained rather than open; this keeps the architecture's computational footprint modest enough to run on commodity branch-office hardware. Second, because throughput scales primarily with block size rather than with difficulty, settlement-cycle length (hourly, end-of-day, or near-real-time) can be tuned independently of the cryptographic security parameter, giving banks a deployment knob that does not trade off against ledger integrity. Third, because the architecture is additive—a tamper-evident proof layer reconciled against, rather than replacing, the core banking system—it can be piloted on a narrow transaction class (e.g., high-value interbranch transfers) with limited disruption to existing operations, before a broader rollout is considered, which also limits the data-protection surface area relevant under regulations such as those analyzed in [19].

ACKNOWLEDGMENT

The author gratefully acknowledges the Southern Regional Representative Office of Agribank and the Ho Chi Minh City University of Industry and Trade (HUIT) for institutional support that made this research possible. Generative-AI tools (Claude, Anthropic) were used to assist with drafting, code implementation, and language editing during manuscript preparation; all technical claims, experimental results, and conclusions were reviewed and verified by the author, who takes full responsibility for the content of this paper, in accordance with IEEE Access policy on the use of generative artificial intelligence in scholarly writing.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

Data Availability: The simulation source code and the raw experimental data (CSV/JSON) underlying every figure and table in this paper are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[2] A. M. Antonopoulos, Mastering Bitcoin: Programming the Open Blockchain, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2017.

[3] Y. Guo and C. Liang, "Blockchain application and outlook in the banking industry," Financial Innovation, vol. 2, no. 1, art. 24, 2016.

[4] R3, Corda: An Introduction (Introductory Whitepaper), R. G. Brown et al., 2016.

[5] E. Androulaki et al., "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," in Proc. 13th EuroSys Conf. (EuroSys '18), 2018, art. 30.

[6] M. Swan, Blockchain: Blueprint for a New Economy. Sebastopol, CA, USA: O'Reilly Media, 2015.

[7] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain Technology Overview," NISTIR 8202, National Institute of Standards and Technology, 2018.

[8] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where Is Current Research on Blockchain Technology?—A Systematic Review," PLOS ONE, vol. 11, no. 10, e0163477, 2016.

[9] F. Casino, T. K. Dasaklis, and C. Patsakis, "A Systematic Literature Review of Blockchain-Based Applications: Current Status, Classification and Open Issues," Telematics and Informatics, vol. 36, pp. 55–81, 2019.

[10] M. Nofer, P. Gomber, O. Hinz, and D. Schiereck, "Blockchain," Business & Information Systems Engineering, vol. 59, no. 3, pp. 183–187, 2017.

[11] P. Treleaven, R. G. Brown, and D. Yang, "Blockchain Technology in Finance," Computer, vol. 50, no. 9, pp. 14–17, 2017.

[12] Y. Chen and C. Bellavitis, "Blockchain Disruption and Decentralized Finance: The Rise of Decentralized Business Models," Journal of Business Venturing Insights, vol. 13, e00151, 2020.

[13] M. Pilkington, "Blockchain Technology: Principles and Applications," in Research Handbook on Digital Transformations, F. X. Olleros and M. Zhegu, Eds. Cheltenham, UK: Edward Elgar Publishing, 2016, pp. 225–253.

[14] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," in Proc. IEEE Int. Congr. Big Data (BigData Congress), 2017, pp. 557–564.

[15] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A Framework for Analyzing Private Blockchains," in Proc. ACM Int. Conf. Manag. Data (SIGMOD '17), 2017, pp. 1085–1100.

[16] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance," in Proc. 3rd USENIX Symp. Operating Syst. Des. Implement. (OSDI '99), 1999, pp. 173–186.

[17] R. Auer and R. Böhme, "The Technology of Retail Central Bank Digital Currency," BIS Quarterly Review, Mar. 2020, pp. 85–100.

[18] Bank for International Settlements, "Proceeding with Caution: A Survey on Central Bank Digital Currency," BIS Papers No. 101, 2020.

[19] M. Finck, "Blockchain and the General Data Protection Regulation: Can Distributed Ledgers Be Squared with European Data Protection Law?" European Parliament, Panel for the Future of Science and Technology (STOA), PE 634.445, 2019.

[20] S. Haber and W. S. Stornetta, "How to Time-Stamp a Digital Document," Journal of Cryptology, vol. 3, no. 2, pp. 99–111, 1991.

[21] A. Back, "Hashcash – A Denial of Service Counter-Measure," Technical Report, 2002.

[22] J. Garay, A. Kiayias, and N. Leonardos, "The Bitcoin Backbone Protocol: Analysis and Applications," in Proc. EUROCRYPT 2015, 2015, pp. 281–310.

[23] I. Eyal and E. G. Sirer, "Majority Is Not Enough: Bitcoin Mining Is Vulnerable," in Proc. Financial Cryptography and Data Security (FC), 2014, pp. 436–454.

[24] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," in Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS), 2016, pp. 3–16.

[25] K. Croman et al., "On Scaling Decentralized Blockchains," in Proc. Financial Cryptography and Data Security (FC) Workshops, 2016, pp. 106–125.

[26] J. Poon and T. Dryja, "The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments," White paper, 2016.

[27] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger," Ethereum Project Yellow Paper, 2014.

[28] V. Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform," Ethereum White Paper, 2014.

[29] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," in Proc. IEEE Symp. Security and Privacy, 2016, pp. 839–858.

[30] G. Zyskind, O. Nathan, and A. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data," in Proc. IEEE Security and Privacy Workshops, 2015, pp. 180–184.

AUTHOR BIOGRAPHY

XUAN PHI NGUYEN is a Senior Expert at the Southern Regional Representative Office of the Vietnam Bank for Agriculture and Rural Development (Agribank), Ho Chi Minh City, Vietnam, and is also affiliated with the Ho Chi Minh City University of Industry and Trade (HUIT), Ho Chi Minh City, Vietnam. He holds a Ph.D. and his research interests include blockchain and distributed ledger technologies, financial technology (fintech), and applications of artificial intelligence in the banking sector.