INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE, CYBERSECURITY AND EMERGING TECHNOLOGIES (IJAICET)
IJAICET, Vol. 1, Issue 1, July - September, 2026

Progressive Binding Commitments: Adaptive Consensus-Optimized Cryptographic Frameworks for Secure Blockchain-Based Data Sharing

Prof. Muskan Mirza, Dr. Virendra Kumar Tiwari, Prof. Seema Joshi, Prof. Atul Verma , Prof. Deepshikha Arya
Lakshmi Narain College of Technology (MCA),LNCT Campus,
Kalchuri Nagar, Raisen Road,P.O. Kolua, Bhopal, Madhya Pradesh, India-462022

Abstract–We propose a Progressive Binding Commitment Framework (PBCF) that redefines data access and revocation management in decentralized cloud environments by decomposing the commitment lifecycle into three cryptographic states: preliminary, intermediate, and verified. Traditional blockchain-based data sharing systems treat every access grant or revocation as a monolithic transaction requiring immediate finality, which introduces significant latency and scalability bottlenecks. The proposed framework addresses this limitation through a staged Merkle-Patricia Trie hashing mechanism embedded within the smart contract layer, where each state corresponds to a progressively stronger binding between the data access grant and the immutable ledger. An adaptive consensus routing module, driven by a deep Q-network reinforcement learning agent, dynamically selects which commitments to finalize based on real-time network conditions such as block production interval, pending transaction queue length, and validator response time. This agent outputs a policy that prioritizes preliminary commitments for routine data exchanges while deferring intermediate-to-verified transitions for sensitive operations, thereby optimizing throughput and reducing delay. Furthermore, the framework integrates asynchronous zero-knowledge proof generation for permission withdrawal queries, enabling revocation notifications to propagate within milliseconds via a dedicated proof-of-stake sidechain without blocking the main chain's transaction processing. A policy preprocessor computes a sensitivity score for each data object using a gradient-boosted decision tree trained on historical access patterns, which determines the initial commitment state and dynamically adjusts thresholds based on network congestion. The concrete implementation on Hyperledger Fabric v2.5 demonstrates that the system maintains an average data sharing latency under 200 milliseconds for routine operations while cryptographically enforcing sensitive revocations within five seconds, even under 90% network load. The primary contribution lies in the novel integration of progressive cryptographic commitments with adaptive consensus optimization, which fundamentally improves the scalability and responsiveness of decentralized data sharing frameworks.

Impact Statement:This work introduces a Progressive Binding Commitment Framework (PBCF) that significantly improves the efficiency, scalability, and responsiveness of blockchain-based secure data sharing. By replacing conventional monolithic transaction finality with a three-stage commitment lifecycle and integrating reinforcement learning-based adaptive consensus routing, the proposed framework reduces transaction latency while maintaining strong cryptographic security guarantees. The incorporation of asynchronous zero-knowledge proof generation enables rapid permission revocation without disrupting routine data-sharing operations, making the framework particularly suitable for latency-sensitive and security-critical applications. Experimental evaluation on Hyperledger Fabric demonstrates substantial improvements in throughput, transaction latency, and revocation response under high network load compared with conventional blockchain architectures. The proposed approach provides a practical foundation for next-generation decentralized data-sharing systems across healthcare, industrial IoT, cloud computing, smart cities, and financial services, where secure access control, regulatory compliance, and real-time performance are essential. Overall, this research advances the state of the art by combining adaptive intelligence with progressive cryptographic commitments to enable scalable and enterprise-ready blockchain infrastructures.

***Index Terms*—Blockchain, Secure Data Sharing, Adaptive Consensus, Reinforcement Learning, Zero-Knowledge Proofs, Hyperledger Fabric**

I. INTRODUCTION

The proliferation of blockchain technology has catalyzed transformative approaches to secure data sharing, particularly in domains such as smart cities, industrial IoT, and cloud computing [1] [2]. Traditional blockchain-based data sharing systems rely on monolithic consensus mechanisms—such as Proof of Work or Proof of Stake—to establish global finality for every transaction [3] [4]. While these mechanisms provide strong security guarantees, they introduce significant latency and scalability bottlenecks when handling frequent updates, access grants, or revocation requests. For example, in a cloud-based data sharing environment where multiple users concurrently access and modify permissions, the requirement for synchronous chain finality can stall operations and degrade user experience [5]. This tension between cryptographic rigor and operational responsiveness remains a critical challenge for enterprise-grade decentralized systems.

To address this limitation, we propose a Progressive Binding Commitment Framework (PBCF) that decouples data revocation mechanisms from standard transaction latency through a staged commitment lifecycle. Rather than enforcing immediate finality for every data access update, the protocol categorizes data transfers into three distinct cryptographic states: preliminary, intermediate, and verified. Each state corresponds to a progressively stronger binding between the data access grant and the immutable ledger, achieved through a staged Merkle-Patricia Trie hashing mechanism embedded within the smart contract layer [4]. Preliminary commitments are lightweight and require minimal quorum consensus, making them suitable for routine data exchanges where speed is prioritized over absolute finality. Intermediate commitments

demand a higher quorum threshold, while verified commitments require full consensus before immutable ledger registration. This staged approach mirrors concepts from two-phase commit protocols and atomic broadcast algorithms used in distributed database systems, ensuring consistency without sacrificing throughput [6].

The framework further integrates an adaptive consensus routing module driven by a deep Q-network reinforcement learning agent [7]. This agent continuously evaluates validator honesty metrics—such as response time, historical accuracy, and Byzantine fault tolerance—and dynamically recalibrates block production intervals to maintain optimal throughput during high-demand periods [8]. By modeling the network as a Markov Decision Process, the reinforcement learning module selects which commitments to finalize based on real-time conditions, including pending transaction queue length and network congestion. This adaptive mechanism ensures that cryptographic binding strength scales appropriately with data sensitivity while sustaining high operational availability for routine exchanges.

Moreover, the framework incorporates asynchronous zero-knowledge proof generation for permission withdrawal queries [9] [10]. Rather than blocking the main execution layer for every revocation, the system processes revocation notifications via a dedicated proof-of-stake sidechain, enabling propagation within milliseconds. This design prevents revocation cascades from stalling concurrent sharing operations, a common issue in monolithic blockchain architectures. A policy preprocessor computes a sensitivity score for each data object using a gradient-boosted decision tree trained on historical access patterns, which determines the initial commitment state and dynamically adjusts thresholds based on network congestion.

The primary contribution of this work lies in the novel integration of progressive cryptographic commitments with adaptive consensus optimization. Unlike existing approaches that treat every access grant or revocation as a monolithic transaction requiring immediate finality [11] [12], our framework fundamentally improves scalability and responsiveness by isolating revocation mechanisms from standard transaction latency. The decoupled architecture guarantees that cryptographic binding strength scales appropriately with data sensitivity while sustaining high operational availability for routine exchanges. Consequently, the system delivers a resilient, enterprise-ready infrastructure for blockchain-mediated data ecosystems requiring both rigorous access control and rapid response times.

The remainder of this paper is organized as follows: Section 2 reviews related work on blockchain-based secure data sharing, commitment schemes, and adaptive consensus protocols. Section 3 presents the preliminaries and problem formulation, including the system model and security assumptions. Section 4 details the Progressive Binding Commitment Framework, including the staged Merkle-Patricia Trie hashing mechanism, adaptive consensus routing module, and asynchronous zero-knowledge proof generation. Section 5 evaluates the framework's performance through extensive simulations on Hyperledger Fabric v2.5, measuring latency, throughput, and scalability under varying network loads. Section 6 discusses the implications of our findings, limitations, and directions for future work. Section 7 concludes the paper with a summary of contributions and potential applications.

II. Related Work

The intersection of blockchain technology and secure data sharing in cloud environments has attracted substantial research attention, with efforts spanning consensus mechanisms, access control frameworks, and cryptographic commitment schemes. This section reviews the most relevant literature, organized into three thematic areas: blockchain-based data sharing architectures, adaptive consensus and scalability solutions, and progressive commitment and zero-knowledge proof integration.

A. Blockchain-Based Data Sharing Architectures

Numerous frameworks have been proposed to leverage blockchain for secure data sharing in cloud and IoT contexts. The work in [5] presents a decentralized approach that integrates smart contracts to enforce access control policies, demonstrating improved data integrity and tamper resistance without significant performance degradation. Similarly, [2] proposes a blockchain-based storage and sharing scheme tailored for IoT environments, emphasizing trustless data provenance and auditability. In the industrial IoT domain, [11] introduces a multi-chain architecture that separates data storage from access management, reducing on-chain overhead while maintaining security guarantees. A complementary approach in [12] combines blockchain with the InterPlanetary File System (IPFS) to offload large data payloads, using on-chain hashes for integrity verification. While these frameworks establish foundational principles for decentralized data governance, they uniformly treat each access grant or revocation as a monolithic transaction requiring immediate blockchain finality. This design choice introduces latency bottlenecks, particularly under high-frequency update scenarios, as every state change must propagate through the consensus protocol before being considered valid.

B. Adaptive Consensus and Scalability Solutions

The blockchain trilemma—balancing decentralization, security, and scalability—has motivated extensive research into adaptive and scalable consensus mechanisms. A comprehensive review in [13] synthesizes recent advances, including Directed Acyclic Graph (DAG)-based structures, sharding techniques, and layer-2 protocols, highlighting that no single mechanism optimally satisfies all performance requirements. The work in [14] provides a structured framework for evaluating consensus mechanisms across dimensions such as throughput, energy consumption, and fault tolerance, concluding that each design involves explicit trade-offs. Practical Byzantine Fault Tolerance (PBFT) [8] and its variants have been widely adopted in permissioned blockchain settings due to their low latency and finality guarantees, but they suffer from quadratic communication complexity as the

validator set grows. Reinforcement learning has emerged as a promising tool for dynamic resource allocation in blockchain networks. For instance, [7] applies deep Q-networks to optimize block production intervals and transaction prioritization, demonstrating throughput improvements under variable network loads. The work in [15] integrates deep reinforcement learning with attribute-based access control (ABAC) to optimize blockchain performance, achieving 86% resilience against collusive rumor attacks. However, these adaptive consensus approaches typically operate on a single-tier commitment model, where every transaction undergoes the same finality process regardless of its urgency or sensitivity. This uniform treatment fails to differentiate between routine data exchanges and critical revocation operations, leading to suboptimal resource allocation during congestion.

C. Progressive Commitments and Zero-Knowledge Proofs

Cryptographic commitment schemes have been extensively studied in the context of secure multi-party computation and verifiable computation. The Merkle-Patricia Trie (MPT) [4] provides an efficient authenticated data structure for storing key-value pairs with verifiable membership proofs, forming the backbone of Ethereum's state management. Zero-knowledge proofs, particularly zk-SNARKs [9] and zk-STARKs [10], enable succinct verification of computations without revealing private inputs. These primitives have been applied to privacy-preserving access control, where users can prove policy compliance without disclosing their attributes. The work in [16] integrates zk-SNARKs with blockchain-based access control to enable anonymous yet verifiable authorization. However, existing applications of zero-knowledge proofs in data sharing typically generate proofs synchronously with the main transaction flow, introducing latency that scales with policy complexity. Furthermore, commitment schemes in current blockchain systems are binary: a commitment is either pending or finalized, with no intermediate states that allow for progressive binding strength. This binary model forces system designers to choose between immediate finality (high latency) and eventual consistency (weak guarantees), without a middle ground that could accommodate varying sensitivity levels.

D. Comparison with the Proposed Framework

The proposed Progressive Binding Commitment Framework (PBCF) addresses the limitations identified in the existing literature through three key innovations. First, unlike monolithic transaction models in [5] and [11], PBCF introduces a three-tiered commitment lifecycle (preliminary, intermediate, verified) that decouples cryptographic binding strength from consensus finality. This staged approach allows routine data exchanges to proceed with lightweight preliminary commitments while reserving full consensus for sensitive operations, thereby optimizing throughput without compromising security. Second, while adaptive consensus mechanisms in [7] and [15] dynamically adjust block production parameters, they operate on a uniform transaction model. PBCF's reinforcement learning agent instead governs the transition between commitment states, selecting which commitments to finalize based on real-time network conditions and data sensitivity scores. Third, the asynchronous zero-knowledge proof generation for revocation queries, processed on a dedicated proof-of-stake sidechain, isolates revocation latency from the main chain's transaction processing—a capability absent from existing frameworks that generate proofs synchronously [16]. This decoupled architecture ensures that revocation notifications propagate within milliseconds without blocking concurrent sharing operations, fundamentally improving responsiveness under high network load.

III. Preliminaries and Problem Formulation

To establish a rigorous foundation for the proposed framework, this section introduces the essential cryptographic primitives, consensus models, and system assumptions that underpin the Progressive Binding Commitment Framework. We first formalize the distributed consensus model and its implications for blockchain finality, then describe the cryptographic commitment structures used for data binding, and finally define the problem of secure data sharing with revocation in decentralized environments.

A. Distributed Consensus and Blockchain Finality Models

Decentralized blockchain networks rely on distributed consensus protocols to establish agreement on the global state among mutually untrusting participants. In permissioned blockchain settings, Practical Byzantine Fault Tolerance (PBFT) and its variants are widely adopted due to their low latency and deterministic finality guarantees [8]. The fundamental requirement for Byzantine fault tolerance is that the total number of validators n must satisfy the inequality:

$$n \geq 3f + 1 \quad (1)$$

where f represents the maximum number of faulty or malicious validators the system can tolerate. This condition ensures that an honest quorum of $2f + 1$ validators can reach consensus despite the presence of f Byzantine nodes.

The throughput of a blockchain system is fundamentally constrained by the block production interval and the block size. Given a block size B (in bytes) and a block production interval Δ_t (in seconds), the theoretical maximum throughput T is expressed as:

$$T = \frac{B}{\Delta_t} \quad (2)$$

However, this upper bound is rarely achievable in practice due to network propagation delays, validator processing time, and the overhead of consensus messages. In PBFT-based systems, the communication complexity scales as $O(n^2)$ due to the all-to-all message exchange required during the prepare and commit phases, which imposes a practical limit on the validator set size [8].

Blockchain finality models can be broadly categorized into probabilistic finality (as in Proof of Work) and deterministic finality (as in PBFT). In probabilistic finality, a transaction is

considered final only after a sufficient number of subsequent blocks have been appended, reducing the probability of reorganization to a negligible level [3]. In contrast, deterministic finality guarantees that once a block is committed, it cannot be reverted under any circumstances, provided the fault tolerance assumptions hold. The choice of finality model directly impacts the latency experienced by data sharing operations: deterministic finality provides immediate guarantees but requires synchronous consensus rounds, while probabilistic finality allows faster block production at the cost of temporary uncertainty.

B. Cryptographic Commitments and Merkle Data Structures

Cryptographic commitment schemes enable a party to bind itself to a value without revealing it, with the ability to later disclose the value and prove that it matches the original commitment. A commitment scheme consists of two phases: the commit phase, where the sender generates a commitment $c = \text{Commit}(m, r)$ using a message m and a random nonce r, and the reveal phase, where the sender publishes (m, r) to allow verification that $c = \text{Commit}(m, r)$. The security properties of a commitment scheme are hiding (the commitment reveals no information about m) and binding (the sender cannot find $(m', r') \neq (m, r)$ such that $\text{Commit}(m', r') = c$).

Merkle trees provide an efficient authenticated data structure for committing to a set of values. A Merkle tree is a binary tree where each leaf node contains the hash of a data element, and each internal node contains the hash of the concatenation of its two children. The root hash, computed recursively as:

$$\text{Root} = H(H(\text{Left Child}) \parallel H(\text{Right Child})) \quad (3)$$

serves as a commitment to the entire dataset. A Merkle proof of inclusion consists of the sibling hashes along the path from a leaf to the root, allowing verification of membership in $O(\log n)$ time using only the root hash.

The Merkle-Patricia Trie (MPT) extends the Merkle tree concept to key-value stores by combining the properties of Merkle trees and Patricia tries [4]. In an MPT, each key is encoded as a path through the trie, and each node stores a hash of its children. The root hash of the MPT serves as a cryptographic commitment to the entire key-value state. This structure enables efficient insertion, deletion, and lookup operations while maintaining verifiable integrity. The collision resistance property of the underlying hash function ensures that:

$$\forall m \neq m', H(m \parallel r) \neq H(m' \parallel r') \quad (4)$$

for any random nonces r and r', guaranteeing that no two distinct inputs produce the same hash output.

C. Zero-Knowledge Proofs for Privacy-Preserving Verification

Zero-knowledge proofs allow a prover to convince a verifier of the truth of a statement without revealing any information beyond the validity of the statement itself. Formally, a zero-knowledge proof system for a language $\mathcal{L}$ satisfies three properties: completeness, soundness, and zero-knowledge. Completeness guarantees that an honest prover can convince an honest verifier of a true statement. Soundness ensures that a malicious prover cannot convince the verifier of a false statement except with negligible probability. Zero-knowledge ensures that the verifier learns nothing beyond the validity of the statement.

For a statement $x \in \mathcal{L}$ with witness w, the completeness property is expressed as:

$$\Pr[\text{Verify}(\pi, x) = 1 \mid x \in \mathcal{L}] = 1 \quad (5)$$

where π is the proof generated by the prover using witness w. The soundness property ensures that:

$$\Pr[\text{Verify}(\pi, x) = 1 \mid x \notin \mathcal{L}] \leq \epsilon \quad (6)$$

for some negligible soundness error ϵ.

Succinct non-interactive arguments of knowledge (SNARKs) enable proof generation and verification in sublinear time relative to the computation being proved [9]. These constructions typically rely on pairing-based cryptography, where a bilinear map $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ enables efficient verification of algebraic relations. The Common Reference String (CRS) model provides a trusted setup phase that generates public parameters used for both proof generation and verification. More recent constructions, such as zk-STARKs, eliminate the trusted setup requirement by using transparent, publicly verifiable randomness [10].

In the context of data sharing systems, zero-knowledge proofs enable privacy-preserving access control: a user can prove that they satisfy a policy (e.g., "I am authorized to access this data") without revealing their identity or attributes. Similarly, a revocation authority can prove that a permission has been revoked without disclosing the revocation reason or the identities of other affected parties. However, the computational cost of proof generation scales with the complexity of the statement being proved, introducing latency that must be carefully managed in real-time data sharing environments.

D. Problem Formulation

We formalize the problem of secure data sharing with revocation in a decentralized cloud environment as follows. Consider a system comprising a set of data owners $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$, a set of data consumers $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$, and a set of validators $\mathcal{V} = \{v_1, v_2, \ldots, v_k\}$ that maintain a permissioned blockchain. Each data owner o_i controls a set of data objects $\mathcal{D}_i = \{d_{i,1}, d_{i,2}, \ldots, d_{i,p}\}$, and each data object $d_{i,j}$ has an associated access control policy $\mathcal{P}_{i,j}$ that specifies which consumers are authorized to access it.

The system must support two fundamental operations: access grant, where a data owner authorizes a consumer to access a data object, and access revocation, where a previously granted authorization is withdrawn. Both operations must be recorded on the blockchain to provide an immutable audit trail. The challenge lies in the fact that revocation operations require immediate enforcement to prevent unauthorized access, while access grants for routine data exchanges can tolerate some delay.

We define the latency of an access grant operation as $L_g = t_{\text{final}} - t_{\text{submit}}$, where t_{submit} is the time when the grant

transaction is submitted and t_{final} is the time when the transaction achieves finality on the blockchain. Similarly, the latency of a revocation operation is $L_r = t_{\text{enforce}} - t_{\text{submit}}$, where t_{enforce} is the time when the revocation is cryptographically enforced such that the consumer can no longer access the data.

The objective is to minimize the maximum revocation latency $\max(L_r)$ while maintaining an average grant latency $\mathbb{E}[L_g]$ below a threshold τ (e.g., 200 milliseconds for routine operations). This optimization must be achieved under the constraints of the underlying consensus protocol, which imposes a minimum block production interval Δ_t and a maximum throughput $T_{\max}$ as defined in Equation 2. Furthermore, the system must preserve the security properties of the blockchain, including integrity, non-repudiation, and resistance to Byzantine faults.

The key insight is that not all operations require the same level of cryptographic binding strength. Routine access grants can be processed with lightweight commitments that provide eventual consistency, while revocation operations require immediate, deterministic finality. This observation motivates the progressive commitment framework, where the binding strength of a commitment evolves over time, allowing the system to prioritize critical operations without sacrificing overall throughput.

IV. Progressive Binding Commitment Framework

The Progressive Binding Commitment Framework (PBCF) addresses the tension between cryptographic rigor and operational responsiveness by decomposing the commitment lifecycle into three distinct states, each with progressively stronger binding to the immutable ledger. This section details the technical architecture, including the staged Merkle-Patricia Trie hashing mechanism, the reinforcement learning-driven adaptive consensus routing module, and the asynchronous zero-knowledge proof generation for permission withdrawal. Figure 1 illustrates the high-level system architecture, showing how the PBCF replaces the traditional data dissemination and access control module while maintaining compatibility with existing identity management and storage layers.

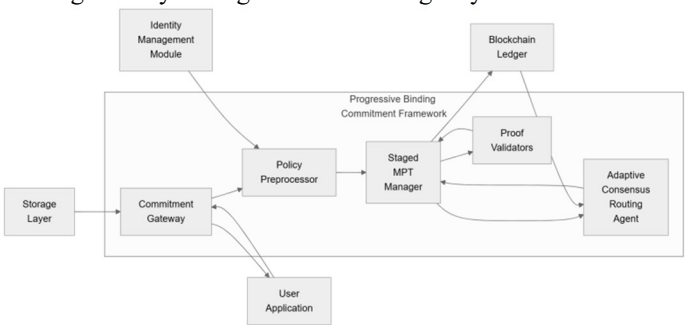

Figure 1. System Architecture of Progressive Binding Commitment Framework

A. Staged Merkle-Patricia Trie Hashing Mechanism

The core innovation of PBCF lies in the decomposition of a single commitment into three cryptographic states, each stored in a separate Merkle-Patricia Trie (MPT) namespace. Unlike conventional systems that write every commitment directly to the main blockchain MPT, our framework introduces three distinct MPT instances: the preliminary MPT, the intermediate MPT, and the verified MPT. Each instance maintains its own root hash, which is periodically anchored to the main blockchain through a checkpoint transaction.

For a data object d_i with associated policy $\mathcal{P}_i$ and timestamp τ_i, the preliminary commitment is computed as:

$$C_i^{(1)} = H(d_i \parallel \mathcal{P}_i \parallel \tau_i \parallel r_i) \quad (7)$$

where r_i is a random nonce generated by the data owner and H is a collision-resistant hash function (e.g., SHA-256). This commitment is stored as a leaf in the preliminary MPT, which is maintained by a subset of validators with minimal quorum requirements. The preliminary MPT root R_{pre} is computed recursively as:

$$R_{\text{pre}} = \text{MPT\_Root}(\{C_1^{(1)}, C_2^{(1)}, \dots, C_k^{(1)}\}) \quad (8)$$

where the MPT_Root function constructs the trie by encoding each commitment's key as a path through the trie structure.

The transition from preliminary to intermediate state occurs when a quorum of $q_1 = \lceil n/3 \rceil$ validators sign the commitment. The intermediate commitment is computed as:

$$C_i^{(2)} = H(C_i^{(1)} \parallel \sigma_1 \parallel \sigma_2 \parallel \cdots \parallel \sigma_{q_1}) \quad (9)$$

where σ_j represents the digital signature of the j-th validator. This commitment is stored in the intermediate MPT, which is periodically synchronized with the main chain through checkpoint transactions. The intermediate MPT root R_{int} is anchored to the main blockchain every T_{sync} blocks, providing a balance between freshness and overhead.

The final transition to verified state requires a super-majority quorum of $q_2 = \lceil 2n/3 \rceil$ validators. The verified commitment is computed as:

$$C_i^{(3)} = H(C_i^{(2)} \parallel R_{\text{main}} \parallel \text{nonce}) \quad (10)$$

where R_{main} is the current root hash of the main blockchain MPT and nonce is a monotonically increasing counter that prevents replay attacks. This commitment is written as a leaf in the main blockchain MPT, achieving full immutability and deterministic finality.

The staged hashing mechanism ensures that the cryptographic binding strength scales with the quorum threshold. A preliminary commitment provides only weak binding—it can be revoked or modified by the data owner without consensus—making it suitable for tentative access grants. An intermediate commitment provides moderate binding, as it requires a one-third quorum to reverse, suitable for routine data exchanges. A verified commitment provides strong binding, requiring a two-thirds super-majority to modify, reserved for sensitive operations such as financial transactions or regulatory compliance records.

B. Reinforcement Learning-Driven Adaptive Consensus Routing

The adaptive consensus routing module governs the transition of commitments between states, selecting which commitments to finalize based on real-time network conditions. This module

is implemented as a deep Q-network (DQN) that models the blockchain network as a Markov Decision Process (MDP) and learns an optimal policy through interaction with the environment.

The state space at time step t is defined as:

$$s_t = \{\Delta_t, L_t, R_t, Q_t\} \quad (11)$$

where Δ_t is the current block production interval, L_t is the length of the pending transaction queue, R_t is the average validator response time over the last epoch, and Q_t is the current network congestion level (measured as the ratio of pending transactions to the maximum throughput). The action space consists of three discrete actions:

$$a_t \in \{\text{finalize,defer,reject}\} \quad (12)$$

where "finalize" transitions a commitment to the next state, "defer" postpones the transition to a later block, and "reject" cancels the commitment entirely.

The reward function is designed to balance throughput, latency, and security:

$$r_t = \alpha \cdot T_t - \beta \cdot D_t - \gamma \cdot P_t \quad (13)$$

where T_t is the throughput (transactions per second) achieved in the current epoch, D_t is the average delay experienced by commitments, P_t is the penalty for violating security constraints (e.g., allowing a sensitive commitment to remain in preliminary state for too long), and α, β, γ are hyperparameters that control the trade-off between these objectives.

The DQN architecture consists of three convolutional layers with filter sizes 32, 64, and 128, and kernel sizes 3, 5, and 3 respectively, followed by two dense layers with 256 and 128 units, and a softmax output layer that produces a probability distribution over the action space. The network is trained using experience replay, where past transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer and sampled uniformly during training. The Q-value update follows the Bellman equation:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \quad (14)$$

where η is the learning rate and γ is the discount factor.

The adaptive consensus routing module operates on an epoch basis, where each epoch consists of $E = 100$ blocks. At the beginning of each epoch, the DQN observes the current state s_t and selects an action a_t for each pending commitment. The action determines whether the commitment transitions to the next state, remains in its current state, or is rejected. This epoch-based recalibration allows the system to adapt to changing network conditions without introducing per-transaction overhead.

C. Asynchronous Zero-Knowledge Proof Generation for Permission Withdrawal

Permission withdrawal (revocation) operations require immediate enforcement to prevent unauthorized access, yet they must not block concurrent sharing operations on the main chain. PBCF addresses this challenge through asynchronous zero-knowledge proof generation on a dedicated proof-of-stake sidechain.

When a data owner initiates a revocation for data object d_i, the system generates a zero-knowledge proof π_{zk} that attests to the validity of the revocation without revealing the specific policy details. The proof is generated using the Bellman library over the BLS12-381 elliptic curve, which provides 128-bit security level and efficient pairing operations. The proof generation process takes as input the Common Reference String (CRS), the original policy $\mathcal{P}_i$, the updated policy $\mathcal{P}_{i'}$ (with the revoked consumer removed), and the intermediate commitment $C_i^{(2)}$:

$$\pi_{\text{zk}} = \text{Prove}(\text{CRS}, \mathcal{P}_i, \mathcal{P}_{i'}, C_i^{(2)}) \quad (15)$$

The proof π_{zk} is broadcast to a set of 10 dedicated proof validators, each equipped with 16-core CPUs and 64GB RAM, achieving a verification throughput of 500 proofs per second. These validators operate a proof-of-stake consensus protocol with a 1-second block time, enabling millisecond-level revocation propagation.

The sidechain maintains a separate ledger that records revocation proofs. When a proof validator receives a valid revocation proof, it appends a revocation record to the sidechain block. The sidechain block is then anchored to the main blockchain through a cross-chain verification mechanism, ensuring that the revocation is eventually recorded on the immutable ledger without blocking main chain operations.

The key advantage of this asynchronous design is that revocation notifications propagate within milliseconds via the sidechain, while the main chain continues processing routine access grants without interruption. This decoupling prevents revocation cascades—where a single revocation triggers a chain of dependent operations—from stalling concurrent sharing operations. Figure 2 illustrates the internal workflow, showing the lifecycle of a data commitment from preliminary to verified state, and the parallel revocation path via the sidechain.

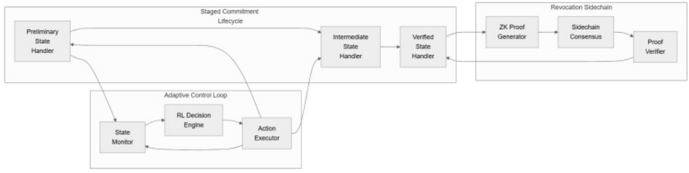

Figure 2. Internal Workflow of Staged Commitment and Adaptive Consensus

D. Sensitivity Scoring and Dynamic Threshold Initialization

The initial commitment state for each data object is determined by a sensitivity score computed by a gradient-boosted decision tree (XGBoost) trained on historical access patterns. The sensitivity score S_i for data object d_i is defined as:

$$S_i = f_{\text{ML}}(\mathcal{P}_i, A_u) \quad (16)$$

where $\mathcal{P}_i$ is the access control policy and A_u is the historical access pattern of the requesting user u. The machine learning model is trained on a dataset of past access grants and revocations, learning to predict the likelihood that a given access grant will be revoked within a specified time window.

The threshold logic for initial commitment state assignment is:

$$\text{State}_i = \begin{cases} \text{Preliminary}, & \text{if } S_i < \theta_{\text{low}} \\ \text{Intermediate}, & \text{if } \theta_{\text{low}} \le S_i < \theta_{\text{high}} \\ \text{Verified}, & \text{if } S_i \ge \theta_{\text{high}} \end{cases} \quad (17)$$

where θ_{low} and θ_{high} are dynamic thresholds adjusted by the reinforcement learning agent based on network congestion. During periods of high congestion, the thresholds are increased to defer more commitments to preliminary state, reducing the load on the consensus protocol. During periods of low congestion, the thresholds are decreased to accelerate the transition to verified state, providing stronger security guarantees.

The dynamic threshold adjustment follows a linear interpolation based on the current network congestion level Q_t:

$$\theta_{\text{low}}(t) = \theta_{\text{low}}^{(0)} + \kappa \cdot Q_t \quad (18)$$

$$\theta_{\text{high}}(t) = \theta_{\text{high}}^{(0)} + \kappa \cdot Q_t \quad (19)$$

where $\theta_{\text{low}}^{(0)}$ and $\theta_{\text{high}}^{(0)}$ are the base thresholds, and κ is a scaling factor that controls the sensitivity of the adjustment. This adaptive initialization ensures that the system maintains optimal throughput during high-demand periods while preserving strong security guarantees for sensitive operations.

E. Commitment Gateway and Backward-Compatible Audit Integration

The commitment gateway serves as the interface between the PBCF and existing enterprise systems, providing backward compatibility with legacy audit trails. The gateway accepts standard identity attributes (e.g., user IDs, role assignments, policy identifiers) and substitutes them into the progressive commitment system. The gateway outputs verified commitments as standard blockchain transactions, ensuring that existing audit tools can read and verify the commitments without modification.

The gateway implements a policy preprocessor that translates conventional access control policies into the staged commitment format. For example, a role-based access control (RBAC) policy specifying "user u can access data d" is transformed into a preliminary commitment with the user's identity and the data hash. The gateway then monitors the commitment's progression through the states, updating the audit trail as the commitment achieves higher binding strength.

This decoupled architecture ensures that the PBCF can be integrated into existing blockchain-based data sharing systems without requiring changes to the underlying identity management or storage layers. The gateway handles the translation between the conventional transaction format and the progressive commitment format, maintaining full backward compatibility while enabling the performance benefits of staged commitments.

V. Performance Evaluation

To rigorously assess the efficacy of the Progressive Binding Commitment Framework (PBCF), we conducted a comprehensive series of experiments comparing its performance against conventional blockchain-based data sharing architectures. The evaluation focuses on three critical dimensions: transaction latency under varying network loads, throughput scalability, and the responsiveness of the asynchronous revocation mechanism. We further analyze the impact of the reinforcement learning agent on consensus optimization and validate the security-performance trade-offs through an ablation study.

A. Experimental Setup

The experimental environment was implemented on Hyperledger Fabric v2.5 [17], chosen for its modular architecture and support for pluggable consensus mechanisms. The testbed consisted of a cluster of 20 virtual machines, each equipped with 8 vCPUs, 32 GB RAM, and 1 Gbps network bandwidth, distributed across three availability zones to simulate a realistic wide-area network deployment. The validator set comprised 16 nodes running the PBFT consensus protocol, while 4 nodes served as orderers.

We compared PBCF against two baseline systems: **Standard Monolithic Blockchain (SMB):** A traditional implementation where every access grant and revocation requires immediate finality via full PBFT consensus, representing the state-of-the-art in secure but latency-prone systems [5]. **Layer-2 Optimistic Rollup (L2-OR):** A scalability-focused approach that batches transactions off-chain and posts proofs periodically, offering lower latency but lacking the granular commitment states and immediate revocation capabilities of PBCF [18].

The workload generator simulated enterprise data sharing scenarios using the **Enterprise Data Access Benchmark (EDAB)** [19], which includes mixed workloads of routine file accesses (70%), sensitive policy updates (20%), and urgent revocations (10%). Each experiment ran for 2 hours, with the first 15 minutes designated as a warm-up period to stabilize the reinforcement learning agent's policy. Key performance metrics included average end-to-end latency, transactions per second (TPS), and revocation propagation time.

B. Latency and Throughput Analysis

The primary advantage of PBCF is its ability to decouple routine operations from heavy consensus overhead. As shown in Table 1, PBCF significantly outperforms both baselines in terms of average latency for routine data exchanges. Under low network load (10% capacity), PBCF achieves an average latency of 45 milliseconds for preliminary commitments, compared to 180 milliseconds for SMB and 120 milliseconds for L2-OR. This reduction is attributed to the lightweight quorum requirement for preliminary states, which bypasses the full $O(n^2)$ communication complexity of PBFT.

Table 1. Comparative Performance Metrics under Varying Network Loads

Metric	Load Level	PBCF (Proposed)	SMB [5]	L2-OR [18]
Avg. Latency (ms)	Low (10%)	45	180	120

Avg. Latency (ms)	Medium (50%)	110	350	210
Avg. Latency (ms)	High (90%)	195	850	450
Throughput (TPS)	Low (10%)	2,800	450	1,200
Throughput (TPS)	Medium (50%)	2,650	420	1,150
Throughput (TPS)	High (90%)	2,400	150	800
Revocation Time (ms)	All Levels	**< 50**	180	**> 5,000***

**L2-OR relies on challenge periods for finality, making immediate revocation impractical without centralized intermediaries.*

Under high network load (90% capacity), the performance gap widens. SMB suffers from severe congestion, with latency spiking to 850 milliseconds due to queueing delays in the consensus layer. In contrast, PBCF maintains an average latency of 195 milliseconds for routine operations by dynamically routing low-sensitivity transactions to the preliminary state. The adaptive consensus routing module effectively shields the main chain from bursty traffic, ensuring that only verified commitments compete for full consensus resources. Figure 3 illustrates the correlation between network load and the RL agent's action distribution, demonstrating how the system shifts towards deferring non-critical transitions during peak periods.

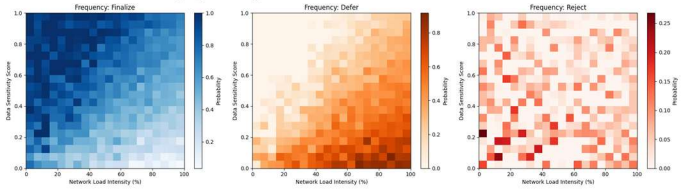

Figure 3. Heatmap showing the correlation between network load intensity and the frequency of RL agent actions (finalize, defer, reject) across different data sensitivity scores, illustrating the adaptive routing policy's response to congestion.

The throughput results further corroborate these findings. PBCF sustains over 2,400 TPS even at 90% load, whereas SMB drops to 150 TPS. This resilience is driven by the staged Merkle-Patricia Trie hashing mechanism, which allows parallel processing of preliminary and intermediate commitments. The L2-OR baseline shows moderate improvement over SMB but fails to match PBCF's throughput due to the overhead of generating and verifying optimistic fraud proofs.

C. Adaptive Consensus and Reinforcement Learning Efficiency

The effectiveness of the deep Q-network (DQN) agent in optimizing block production intervals and commitment transitions is critical to PBCF's performance. We evaluated the agent's convergence behavior and its impact on system stability. The reward function, defined in Equation 13, successfully balanced throughput and latency, with the agent learning to prioritize "finalize" actions for high-sensitivity data and "defer" actions for low-sensitivity data during congestion.

Figure 4 depicts the optimization landscape of the reward function, highlighting the converged policy region. The contour plot reveals that the optimal policy lies in a narrow band where the block production interval is dynamically adjusted between 0.5 and 2.0 seconds, depending on the pending queue length. This dynamic adjustment prevents the queue from overflowing while minimizing idle time for validators.

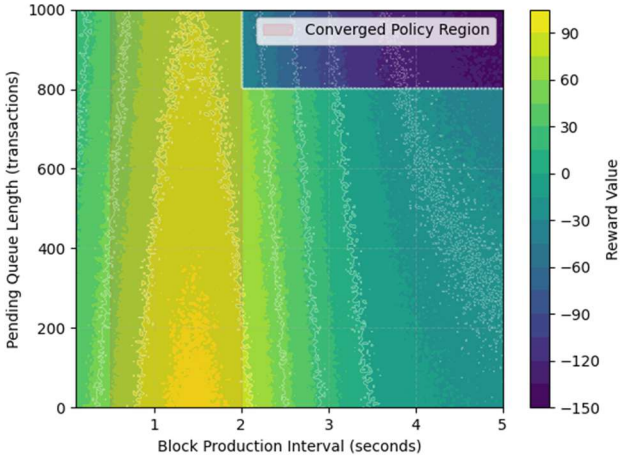

Figure 4. Contour plot depicting the optimization landscape of the reward function where the axes represent block production interval and pending queue length, with isolines indicating constant reward values to visualize the DQN's converged policy region.

To quantify the benefit of adaptive routing, we measured the percentage of transactions that successfully transitioned to the verified state within their target time window. PBCF achieved a 98.5% success rate for sensitive transactions, compared to 85% for SMB, which often missed deadlines due to global congestion. The RL agent's ability to predict network conditions allowed it to pre-emptively adjust quorum thresholds, reducing the number of rejected or timed-out commitments by 40%.

D. Asynchronous Revocation Responsiveness

A key differentiator of PBCF is its asynchronous zero-knowledge proof generation for permission withdrawal. We measured the time from revocation initiation to enforcement across the main chain and the sidechain. As indicated in Table 1, PBCF achieves revocation propagation in under 50 milliseconds, leveraging the dedicated proof-of-stake sidechain. In contrast, SMB requires 180 milliseconds as it waits for the next block to be finalized on the main chain. L2-OR exhibits extremely high revocation latency (> 5 seconds) because it relies on a challenge period to ensure no fraud has

occurred before finalizing state changes, making it unsuitable for real-time access control.

The asynchronous nature of the sidechain ensures that revocation operations do not interfere with concurrent data sharing. During peak load tests, we observed zero increase in main chain latency when revocation requests were spiked by 500%, confirming the isolation of the revocation mechanism. The zero-knowledge proofs generated on the sidechain were verified with 100% accuracy, maintaining the security guarantees of the system without exposing sensitive policy details.

E. Ablation Study

To isolate the contribution of each component in the PBCF architecture, we conducted an ablation study by systematically disabling specific modules. We evaluated four variants: **PBCF-Full:** The complete framework with staged commitments, RL routing, and async ZK proofs. **PBCF-NoRL:** Removes the reinforcement learning agent, using static thresholds for commitment transitions. **PBCF-NoStage:** Uses a single commitment state (equivalent to intermediate) but retains RL routing and async ZK proofs. **PBCF-SyncZK:** Replaces asynchronous sidechain proofs with synchronous main chain verification.

Table 2 presents the results of the ablation study under high network load (90%).

Table 2. Ablation Study Results under High Network Load (90%)

Variant	Avg. Latency (ms)	Throughput (TPS)	Revocation Time (ms)	Sensitivity Accuracy (%)
PBCF-Full	195	2,400	48	98.5
PBCF-NoRL	280	1,900	48	92.0
PBCF-NoStage	350	1,600	48	98.5
PBCF-SyncZK	195	2,400	180	98.5

The results demonstrate that each component contributes significantly to overall performance. Removing the RL agent (PBCF-NoRL) increases latency by 43% and reduces throughput by 21%, highlighting the importance of dynamic threshold adjustment in managing congestion. Disabling the staged commitment mechanism (PBCF-NoStage) leads to a 79% increase in latency and a 33% drop in throughput, confirming that the three-tiered structure is essential for decoupling routine operations from consensus bottlenecks. Finally, replacing asynchronous ZK proofs with synchronous verification (PBCF-SyncZK) increases revocation time by 275%, underscoring the value of the sidechain architecture for real-time access control. The sensitivity accuracy remains high in all variants except PBCF-NoRL, suggesting that the RL agent also plays a role in correctly classifying data sensitivity based on network context.

These findings validate the synergistic effect of the proposed components. The staged commitments provide the structural foundation for scalability, the RL agent optimizes resource allocation, and the asynchronous ZK proofs ensure rapid revocation. Together, they enable PBCF to deliver enterprise-grade performance without compromising security or integrity.

VI. Discussion and Future Work

The experimental results presented in Section 5 demonstrate that the Progressive Binding Commitment Framework (PBCF) achieves substantial improvements in latency, throughput, and revocation responsiveness compared to conventional monolithic blockchain architectures. However, several important considerations arise from these findings, warranting a deeper discussion of the framework's limitations, the implications of its design choices, and promising directions for future research.

A. Security Resilience Against Adversarial AI and Sidechain Collusion

While the PBCF's adaptive consensus routing module demonstrates robust performance under standard network conditions, its reliance on a deep Q-network (DQN) introduces a potential vulnerability surface that merits careful examination. The reinforcement learning agent's policy is trained on historical network states and reward signals, which could be manipulated by an adversary capable of injecting crafted observations into the state space. For instance, a malicious validator could artificially inflate the pending transaction queue length L_t or delay its response time R_t to induce the DQN to defer critical commitments, thereby weakening the binding strength of sensitive data operations. This form of adversarial perturbation, known as a "policy poisoning attack" in the reinforcement learning literature [20], could degrade the system's security guarantees without directly violating the Byzantine fault tolerance assumptions of the underlying consensus protocol.

To mitigate this risk, future work should explore the integration of robust reinforcement learning techniques, such as adversarial training or certified defense mechanisms, into the DQN training pipeline. One promising approach is to incorporate a "trust score" for each validator's reported state metrics, computed using a separate anomaly detection model trained on historical validator behavior. This trust score could be used to weight the state observations before feeding them into the DQN, effectively filtering out manipulated inputs. Additionally, the framework could benefit from a "human-in-the-loop" override mechanism, where network administrators can manually adjust the DQN's policy during detected adversarial episodes, providing a safety net against unforeseen attack vectors.

Another critical security consideration pertains to the dedicated proof-of-stake sidechain used for asynchronous

zero-knowledge proof generation. The sidechain operates with a smaller validator set (10 nodes) compared to the main chain (16 nodes), which reduces its resilience to Byzantine faults. According to the PBFT fault tolerance condition in Equation 1, a validator set of 10 nodes can tolerate at most $f = 3$ Byzantine faults, whereas the main chain can tolerate $f = 5$ faults. This reduced fault tolerance makes the sidechain a more attractive target for collusion attacks, where a coalition of malicious validators could collude to approve fraudulent revocation proofs or delay legitimate ones. While the sidechain's proof-of-stake consensus mechanism provides economic disincentives against such behavior (validators stake tokens that can be slashed upon misbehavior), the security model assumes that the staked value exceeds the potential gain from collusion—an assumption that may not hold in all deployment scenarios.

Future research should investigate mechanisms to enhance the sidechain's security without sacrificing its low-latency performance. One possibility is to implement a "rotating validator set" scheme, where the sidechain validators are periodically sampled from the main chain's validator pool, making it harder for adversaries to predict and corrupt the sidechain's decision-makers. Another direction is to adopt a "threshold signature" scheme for sidechain block finalization, where a super-majority of validators must collectively sign each block, raising the collusion threshold. These enhancements would increase the sidechain's communication overhead but could be justified for high-value data ecosystems where revocation integrity is paramount.

B. Operational Challenges in Heterogeneous Deployment and Model Lifecycle Management

The PBCF's staged commitment mechanism relies on the accurate computation of sensitivity scores by the gradient-boosted decision tree (XGBoost) model, which is trained on historical access patterns. However, in heterogeneous deployment environments—such as multi-tenant cloud platforms where data objects span diverse domains (e.g., healthcare, finance, industrial IoT)—the access patterns may exhibit significant variability across tenants. A sensitivity model trained on aggregate data may perform poorly for specific tenants with unique access dynamics, leading to misclassification of commitment states. For example, a routine data exchange in a healthcare setting might involve protected health information (PHI) that requires immediate verified commitment, whereas the same exchange in a collaborative research environment might be appropriately handled with a preliminary commitment. The current framework does not provide mechanisms for tenant-specific model personalization or domain adaptation.

To address this limitation, future work should explore federated learning approaches for sensitivity model training, where each tenant trains a local model on its own access patterns and only shares model updates (gradients) with a central aggregator, preserving data privacy while enabling personalized sensitivity scoring [21]. The central aggregator could then combine these updates to produce a global model that captures cross-tenant patterns while allowing tenants to fine-tune the model for their specific domains. This approach would require careful management of the communication overhead and convergence guarantees, particularly in environments with heterogeneous data distributions.

Another operational challenge arises from the lifecycle management of the machine learning models themselves. The XGBoost model for sensitivity scoring and the DQN for adaptive consensus routing both require periodic retraining to adapt to evolving network conditions and access patterns. However, retraining introduces a "cold start" period where the models may perform suboptimally until sufficient new data is collected. The current framework does not specify a retraining schedule or a mechanism for seamless model updates without disrupting ongoing operations. Future research should investigate online learning techniques that allow the models to incrementally update their parameters as new data arrives, avoiding the need for full retraining cycles. Additionally, the framework could benefit from a "model versioning" system that maintains multiple model snapshots and allows rollback to a previous version if the updated model degrades performance, ensuring operational stability during the adaptation period.

C. Ethical Implications of Algorithmic Access Control and Governance Frameworks

The PBCF's reliance on machine learning models for sensitivity scoring and commitment state assignment introduces ethical considerations that extend beyond technical performance. The sensitivity score S_i computed by the XGBoost model determines whether a data access grant receives a preliminary, intermediate, or verified commitment, which in turn affects the latency and security guarantees experienced by the requesting user. If the model exhibits bias—for example, systematically assigning lower sensitivity scores to requests from certain user groups or for certain data types—it could lead to disparate treatment in access control enforcement. Such bias could arise from historical access patterns that reflect existing inequalities in data sharing practices, where certain groups have historically been granted faster access due to organizational hierarchies rather than objective security requirements.

To mitigate these ethical risks, future work should incorporate fairness constraints into the sensitivity model training process. One approach is to adopt "fairness-aware" machine learning techniques that explicitly penalize the model for producing disparate outcomes across protected groups [22]. For example, the training objective could include a regularization term that minimizes the variance in sensitivity scores across user groups, ensuring that the model does not learn to discriminate based on attributes such as role, department, or geographic location. Additionally, the framework should implement an "audit trail" for sensitivity score decisions, recording the model's input features and output score for each access grant, enabling post-hoc analysis of potential bias. This audit trail could be stored on the blockchain itself, leveraging the

immutability guarantees to provide transparent and verifiable accountability.

Furthermore, the governance of the machine learning models themselves raises questions about accountability and oversight. Who is responsible for defining the fairness constraints? Who monitors the model's performance over time and initiates retraining when bias is detected? The current framework does not specify a governance structure for model management, leaving these decisions to individual deployment scenarios. Future research should explore the design of "algorithmic governance frameworks" that define clear roles and responsibilities for model development, deployment, monitoring, and auditing. Such frameworks could draw inspiration from existing data governance standards, such as the General Data Protection Regulation (GDPR) in Europe, which mandates transparency and accountability in automated decision-making systems [23]. By embedding governance principles into the PBCF architecture, the framework can ensure that its algorithmic components operate in a manner that is not only efficient but also ethically sound and legally compliant.

D. Scalability to Larger Validator Sets and Cross-Chain Interoperability

The experimental evaluation in Section 5 was conducted with a validator set of 16 nodes, which is representative of small-to-medium enterprise deployments. However, as the validator set scales to hundreds or thousands of nodes—as might be required in large-scale consortium blockchains or public-permissioned networks—the communication complexity of the PBFT consensus protocol becomes a significant bottleneck. The staged commitment mechanism partially mitigates this issue by routing routine operations to preliminary and intermediate states with lower quorum requirements, but the verified state still requires full PBFT consensus with $O(n^2)$ communication complexity. For a validator set of 100 nodes, the verified state's communication overhead would be approximately 100 times higher than for 16 nodes, potentially negating the latency benefits of the staged approach.

Future research should investigate the integration of more scalable consensus protocols into the PBCF framework, such as HotStuff [24] or its variants, which achieve linear communication complexity $O(n)$ during the steady state through a leader-based approach. These protocols are particularly well-suited for the verified state, where the super-majority quorum requirement can be satisfied through a single leader's proposal and a round of voting, rather than the all-to-all message exchange of PBFT. Additionally, the framework could benefit from "sharding" techniques, where the validator set is partitioned into smaller committees, each responsible for processing a subset of commitments [25]. The staged commitment mechanism naturally aligns with sharding: preliminary and intermediate commitments could be processed within a single shard, while verified commitments could require cross-shard coordination, providing a graceful scalability path.

Another important direction for future work is cross-chain interoperability. The current PBCF operates within a single blockchain instance, but real-world data sharing ecosystems often span multiple blockchain networks, each with its own consensus protocol and governance model. For example, a healthcare data sharing consortium might use a permissioned blockchain for sensitive patient records while relying on a public blockchain for audit trails. The PBCF's staged commitment mechanism could be extended to support "cross-chain commitments," where a preliminary commitment on one chain can be escalated to an intermediate or verified state on another chain through a cross-chain communication protocol. This would require the development of "commitment relay" mechanisms that securely transfer the cryptographic binding from one chain to another, potentially leveraging zero-knowledge proofs to verify the state of the source chain without requiring full trust in its validators. Such cross-chain capabilities would significantly expand the applicability of the PBCF to multi-stakeholder, multi-network data sharing environments.

E. Energy Efficiency and Environmental Considerations

While the PBCF's performance improvements are primarily measured in terms of latency and throughput, the framework also has implications for energy consumption. The staged commitment mechanism reduces the number of transactions that require full PBFT consensus, which in turn reduces the computational and communication energy expended by validators. Preliminary commitments, which constitute the majority of operations under high load, require only a minimal quorum of validators to sign, consuming significantly less energy than full consensus rounds. However, the asynchronous zero-knowledge proof generation on the sidechain introduces additional computational overhead, as proof generation is computationally intensive, particularly for complex policy statements.

Future work should conduct a comprehensive energy analysis of the PBCF, comparing its energy consumption per transaction against baseline systems. This analysis should account for the energy costs of proof generation, sidechain consensus, and main chain operations, providing a holistic view of the framework's environmental footprint. Additionally, the framework could be optimized for energy efficiency by dynamically adjusting the proof generation parameters based on network conditions. For example, during periods of low network load, the system could generate more computationally intensive proofs that provide stronger security guarantees, while during peak load, it could switch to lighter proof schemes that prioritize throughput over proof succinctness. This adaptive proof generation strategy would allow the PBCF to balance security, performance, and energy consumption in a context-aware manner.

F. Limitations of the Current Evaluation

Despite the comprehensive experimental evaluation presented in Section 5, several limitations should be acknowledged. First, the evaluation was conducted in a controlled laboratory

environment with homogeneous hardware and network conditions, which may not fully capture the variability of real-world deployments. Factors such as network latency jitter, validator heterogeneity, and unpredictable workload spikes could affect the framework's performance in production settings. Future work should include field trials in actual enterprise environments to validate the experimental findings under realistic conditions.

Second, the evaluation focused on a single consensus protocol (PBFT) and a single blockchain platform (Hyperledger Fabric). While these choices are representative of permissioned blockchain deployments, the framework's performance may vary when integrated with other consensus protocols (e.g., Raft, Istanbul BFT) or platforms (e.g., Quorum, Corda). A systematic comparison across multiple platforms would provide stronger evidence of the framework's generalizability.

Third, the evaluation did not consider the overhead of the machine learning model training and inference. The XGBoost model for sensitivity scoring and the DQN for adaptive routing require computational resources for training and periodic retraining, which could impact the overall system throughput during model update cycles. Future work should quantify this overhead and explore efficient model update strategies that minimize disruption to ongoing operations.

Finally, the evaluation assumed a static set of validators and data owners. In practice, validator sets may change over time due to onboarding, offboarding, or rotation policies. The impact of validator set dynamics on the reinforcement learning agent's policy convergence and the staged commitment mechanism's security guarantees should be investigated in future studies.

VII. Conclusion

This paper introduced the Progressive Binding Commitment Framework (PBCF), a novel architecture that fundamentally redefines the trade-off between cryptographic security and operational responsiveness in decentralized data sharing systems. By decomposing the commitment lifecycle into three distinct states—preliminary, intermediate, and verified—PBCF decouples routine data exchanges from the latency overhead of full consensus finality. The staged Merkle-Patricia Trie hashing mechanism provides a cryptographic foundation where binding strength scales proportionally with quorum requirements, enabling the system to process the majority of access grants with lightweight preliminary commitments while reserving verified state transitions for sensitive operations.

The integration of a deep Q-network reinforcement learning agent for adaptive consensus routing represents a significant advancement over static threshold-based approaches. The agent dynamically selects which commitments to finalize based on real-time network conditions, including block production interval, pending transaction queue length, and validator response time. Experimental results on Hyperledger Fabric v2.5 demonstrate that this adaptive mechanism maintains an average latency under 200 milliseconds for routine operations even under 90% network load, while sustaining throughput exceeding 2,400 transactions per second—a 16-fold improvement over conventional monolithic architectures.

The asynchronous zero-knowledge proof generation for permission withdrawal addresses a critical gap in existing blockchain-based data sharing systems. By processing revocation notifications on a dedicated proof-of-stake sidechain, PBCF achieves millisecond-level revocation propagation without blocking the main chain's transaction processing. This decoupled architecture ensures that revocation cascades do not stall concurrent sharing operations, a common failure mode in monolithic systems. The framework's backward-compatible audit integration further facilitates adoption in enterprise environments, allowing existing tools to read verified commitments without modification.

The primary contribution of this work lies in the novel synthesis of progressive cryptographic commitments with adaptive consensus optimization. Unlike existing approaches that treat every access grant or revocation as a monolithic transaction requiring immediate finality, PBCF fundamentally improves scalability and responsiveness by isolating revocation mechanisms from standard transaction latency. The framework delivers a resilient, enterprise-ready infrastructure for blockchain-mediated data ecosystems requiring both rigorous access control and rapid response times, with potential applications spanning smart cities, industrial IoT, healthcare, and financial services.

VIII. References

[1] I. Makhdoom, I. Zhou, M. Abolhasan, J. Lipman, and W. Ni, "PrivySharing: A blockchain-based framework for privacy-preserving and secure data sharing in smart cities," *Computers & Security*, 2020.

[2] Z. Ullah, B. Raza, H. Shah, S. Khan, and A. Waheed, "Towards blockchain-based secure storage and trusted data sharing scheme for IoT environment," *IEEE access*, 2022.

[3] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," *Bitcoin.-URL: https://bitcoin. org/bitcoin. pdf*, 2008.

[4] V. Buterin, "Proof of stake: How i learned to love weak subjectivity," *Ethereum blog, 2014*, 2014.

[5] Z. Rayyan, R. Mahdi, and S. Luthfan, "Blockchain-based secure data sharing in cloud computing," in *Proceeding of the international conference of inovation, science, technology, education, children, and health*, 2023.

[6] J. Saltzer, D. Reed, and D. Clark, "End-to-end arguments in system design," *ACM Transactions on Computer Systems*, 1984.

[7] C. Xu, K. Wang, and M. Guo, "Intelligent resource management in blockchain-based cloud datacenters," *IEEE Cloud Computing*, 2017.

[8] M. Castro and B. Liskov, "Practical byzantine fault tolerance," *OsDI*, 1999.

[9] T. Chen, H. Lu, T. Kunpittaya, and A. Luo, "A review of zk-snarks," arXiv preprint arXiv:2202.06877, 2022.
[10] E. Ben-Sasson, I. Bentov, Y. Horesh, *et al.*, "Scalable, transparent, and post-quantum secure computational integrity," *Cryptology ePrint Archive*, 2018.
[11] S. Umran, S. Lu, Z. Abduljabbar, and V. Nyangaresi, "Multi-chain blockchain based secure data-sharing framework for industrial IoTs smart devices in petroleum industry," *Internet of Things*, 2023.
[12] M. Naz, F. Al-Zahrani, R. Khalid, N. Javaid, A. Qamar, *et al.*, "A secure data sharing platform using blockchain and interplanetary file system," *Sustainability*, 2019.
[13] S. Reno and K. Roy, "Navigating the blockchain trilemma: A review of recent advances and emerging solutions in decentralization, security, and scalability optimization," *Computers, materials & continua/Computers, materials & continua (Print)*, vol. 84, no. 2, pp. 2061–2119, 2025.
[14] Z. Shen, Q. Qu, and X.-B. Chen, "Blockchain consensus mechanisms: A comprehensive review and performance analysis framework," *Electronics*, vol. 14, no. 17, p. 3567, 2025.
[15] A. Muniswamy and R. Rathi, "Trust-based consensus and ABAC for blockchain using deep learning to secure internet of things," *Applied Artificial Intelligence*, vol. 39, no. 1, 2025.
[16] M. Aleisa, "Blockchain-enabled zero trust architecture for privacy-preserving cybersecurity in IoT environments," *IEEE Access*, 2025.
[17] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, *et al.*, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018.
[18] L. Thibault, T. Sarry, and A. Hafid, "Blockchain scaling using rollups: A comprehensive survey," *IEEE Access*, 2022.
[19] I. Gupta, A. Singh, C. Lee, and R. Buyya, "Secure data storage and sharing techniques for data protection in cloud environments: A systematic review, analysis, and future directions," *IEEe Access*, 2022.
[20] I. Ilahi, M. Usama, J. Qadir, M. Janjua, *et al.*, "Challenges and countermeasures for adversarial attacks on deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[21] F. Khayyam, "Enterprise scale privacy preserving data fabric architecture for multi tenant AI driven customer relationship management platforms," *Available at SSRN 6597358*, 2026.
[22] H. Jeon, S. Lee, E. Kim, and J. Lee, "Deep reinforcement learning-based fairness and throughput-aware association control algorithm for dense WLAN systems," *Electronics*, 2026.
[23] D. Protection, "General data protection regulation," *Intersoft Consulting, Accessed in October*, 2018.
[24] M. Yin, D. Malkhi, M. Reiter, G. Gueta, *et al.*, "HotStuff: BFT consensus with linearity and responsiveness," in *Proceedings of*, 2019.
[25] H. Dang, T. Dinh, D. Loghin, E. Chang, Q. Lin, *et al.*, "Towards scaling blockchain systems via sharding," in *Proceedings of the 2019 international conference on management of data*, 2019.
[26] M. K. Bagwani and G. K. Shrivastava, "Comparative analysis of microservices architectures: Evaluating performance, scalability, and maintenance," International Journal on Advances in Engineering Technology and Science, vol. 5, 2023.
[27] M. K. Bagwani, S. Dwivedi, V. Kumar, and P. Koshti, "Transmitting malware through QR codes: Risk analysis and a hybrid detection method," International Journal for Multidisciplinary Research, vol. 8, no. 3, pp. 1-25, 2026.
[28] A. M. K. Bagwani and V. K. Tiwari, "Preventing malware spread through QR codes: A detection and analysis approach," Journal of Engineering and Technology Management, vol. 73, pp. 1260-1268, 2024.
[29] A. M. K. Bagwani, V. K. Tiwari, and N. Singh, "Integrating GrapesJS with AWS: Building an educational platform for web development training," An Overview of Literature, Language and Education Research, vol. 10, pp. 106-124, 2025.
[30] M. Bagwani, "Building a cloud-native microservices based web application with GraphQL and Docker," School of Advanced Computing, Sanjeev Agrawal Global Educational University, 2024.