BLOCKCHAIN ARCHITECTURE AND PROTOCOL EVOLUTION: A SURVEY OF CONSENSUS, FORKING, AND SCALABILITY CHALLENGES

[1]Maitri Hingu, [2]Kamlendu Pandey
[1]Assistant Professor, [2]Professor
[1,2]Department of Information and Communications Technology
[1,2]Veer Narmad South Gujarat University, Surat, Gujarat, India
[1]maitrikhingu@gmail.com, [2]kamlendup@gmail.com

Abstract: Blockchain systems rely on architectural design choices and consensus protocols to establish decentralized trust in distributed environments. This paper presents a focused survey of blockchain architecture and protocol evolution, emphasizing structural components, peer-to-peer networking, consensus mechanisms, forking models, and security-scalability trade-offs. Core elements such as blocks, cryptographic hashing, distributed ledgers, node roles, transaction propagation, and validation processes are examined to explain how integrity and immutability are maintained. Major consensus mechanisms, including Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), and Proof of Authority (PoA), are comparatively analyzed with respect to decentralization, throughput, finality, energy consumption, and deployment context. The paper also examines blockchain forking as a mechanism for protocol evolution and governance. By distinguishing protocol-level concerns from application-level adoption, this survey provides a technical foundation for evaluating blockchain systems and identifies open challenges in scalability, interoperability, governance, privacy, and sustainable consensus design.

Index Terms **- Blockchain Architecture, Distributed Ledger, Cryptographic Hashing, Peer-to-Peer Network, Consensus Mechanism, Proof of Work, Proof of Stake, PBFT, Proof of Authority, Forking, Scalability, Blockchain Security.**

1. INTRODUCTION

Blockchain technology introduces a decentralized trust model for distributed systems by allowing network participants to maintain a shared ledger without relying on a central authority. A blockchain records transactions in a sequence of cryptographically linked blocks, where each block depends on the integrity of previous blocks. This structure provides transparency, tamper resistance, and fault tolerance, making blockchain suitable for decentralized environments in which participants may not fully trust one another [1–3].

The effectiveness of a blockchain system depends strongly on its architecture and consensus protocol. Structural components such as blocks, hash pointers, peer-to-peer communication, transaction validation, and distributed storage determine how data is recorded and propagated. Earlier structural studies of blockchain have also emphasized architectural configurations, consensus protocols, public and private blockchain models, and forking as key components of blockchain evolution [4]. Consensus mechanisms determine how distributed nodes agree on the current state of the ledger despite delays, failures, or malicious behavior. Early blockchain systems relied heavily on Proof of Work (PoW), which demonstrated the feasibility of decentralized consensus but introduced limitations related to energy consumption and throughput [1–3].

To address these limitations, several alternative consensus mechanisms and permissioned blockchain architectures have emerged. Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), Proof of Authority (PoA), and hybrid approaches attempt to improve scalability, finality, energy efficiency, or enterprise suitability [2, 5, 6]. Each mechanism offers a different trade-off among decentralization, performance, security, and trust assumptions.

Protocol evolution is another important dimension of blockchain architecture. Forks may occur because of software upgrades, governance disagreement, or temporary network divergence. While accidental forks are usually resolved by consensus rules, intentional forks such as soft forks and hard forks can change protocol behavior and affect compatibility, governance, and community trust [2, 3, 7].

Although many studies examine blockchain applications, a clear architectural perspective is required to understand why blockchain systems behave differently across environments. This paper therefore focuses on protocol-level design rather than application-level adoption. It surveys blockchain architecture, node roles, consensus mechanisms, forking models, and security-scalability considerations. The objective is to provide a compact but technically meaningful foundation for researchers, developers, and system architects.

The reviewed literature was categorized according to architectural relevance, including data structure, networking, consensus, forking, security, and scalability concerns. Application-focused studies were excluded from the core analysis unless they provided direct insight into architectural design, protocol behavior, performance evaluation, or deployment constraints.

Table 1. Layered view of blockchain architecture.

Layer	Main Function	Key Design Concern
Data layer	Stores transactions, blocks, times-tamps, and hash links	Integrity, immutability, storage overhead
Network layer	Propagates transactions and blocks among nodes	Latency, bandwidth, peer discovery
Consensus layer	Establishes agreement on valid blocks and ledger state	Security, finality, decentralization, scalability
Contract layer	Executes programmable rules and decentralized logic	Correctness, security verification, gas/resource cost
Application layer	Provides user-facing decentralized services	Usability, interoperability, compliance

2. BLOCKCHAIN ARCHITECTURE OVERVIEW

Blockchain architecture can be understood as a layered structure that combines data organization, cryptographic linking, peer-to-peer communication, consensus, and programmable execution. At the data layer, transactions are grouped into blocks. At the network layer, nodes propagate transactions and blocks. At the consensus layer, participants agree on the valid state of the ledger. In systems supporting smart contracts, a contract layer executes predefined logic, while the application layer exposes blockchain functionality to users and services [2, 3, 8].

Prior work on blockchain structural dynamics describes blocks, peer-to-peer networking, nodes, transactions, smart contracts, wallets, and consensus as core architectural elements that determine how blockchain systems maintain integrity and decentralized operation [4].

This layered perspective is useful because most blockchain limitations arise from interactions among layers. For example, high transaction volume affects network propagation, consensus latency, and storage growth. Similarly, security depends not only on cryptographic hashing but also on node behavior, validator incentives, and protocol governance.

2.1 Blocks, Hashing, and Distributed Ledger

A blockchain is structured as a chain of blocks connected through cryptographic hash references. Each block normally contains transaction data, a timestamp, a nonce or validation metadata, and the hash of the preceding block [1–3]. If the contents of a block are altered, its hash changes, thereby breaking the relationship with subsequent blocks. This property makes unauthorized modification detectable and supports the immutability of the ledger.

The distributed ledger is replicated across participating nodes rather than stored in a single central database. Replication improves availability and fault tolerance because the failure or compromise of one node does not destroy the entire ledger [3]. However, distributed replication also creates challenges related to storage cost, synchronization, and consistency across nodes [9–11]. There-fore, modern blockchain research often explores storage optimization, sharding, pruning, off-chain storage, and layer-2 mechanisms. Figure 1 illustrates how valid blocks are linked using current and previous block hashes.

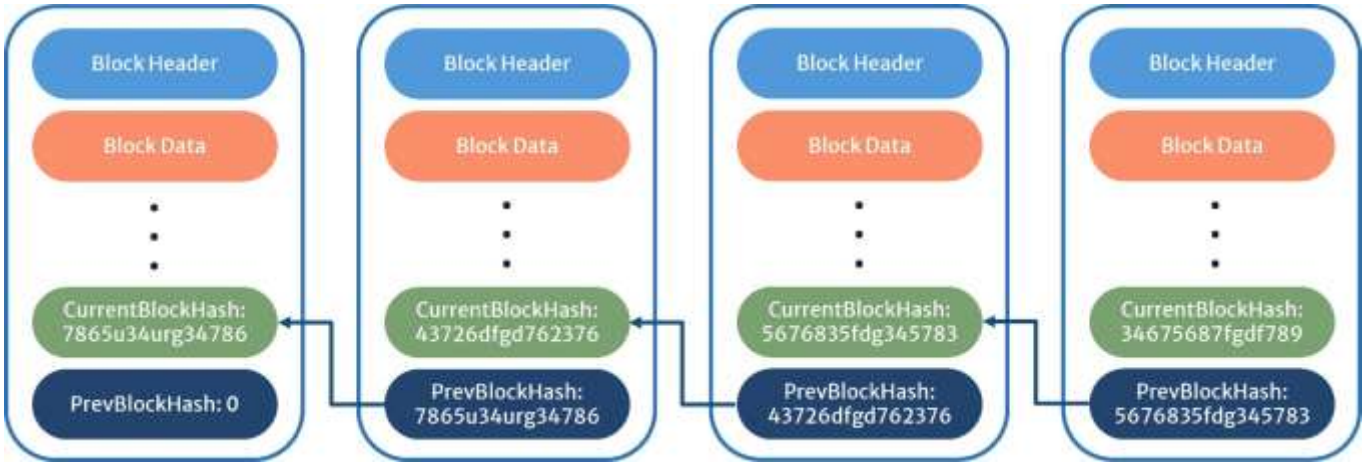

Figure 1. Valid blockchain structure showing linkage between current block hash and previous block hash.

Figure 2 shows how modification of block data disrupts hash continuity and breaks the validity of the chain.

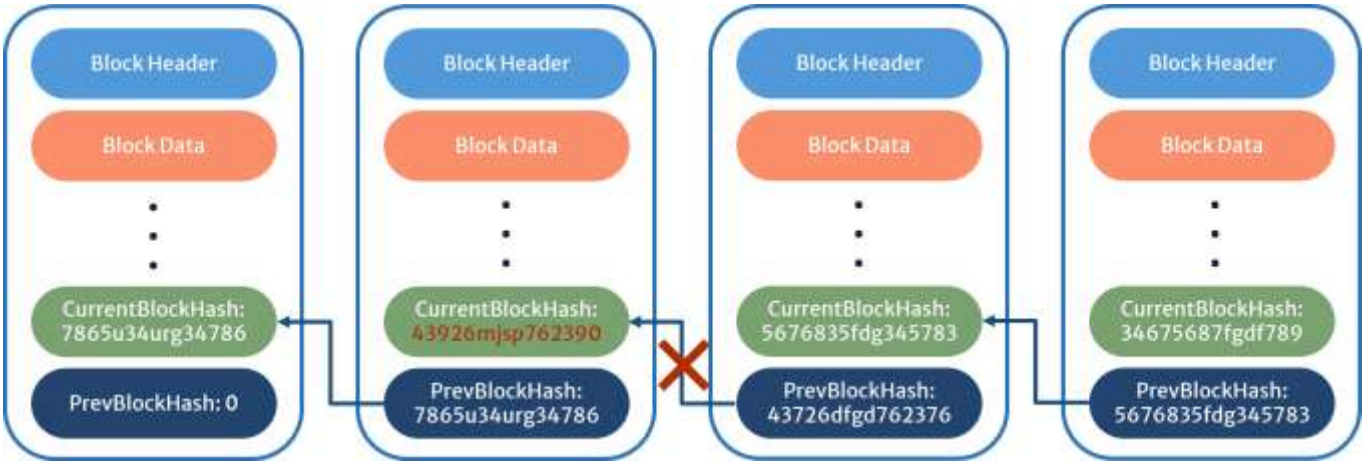

Figure 2. Effect of data tampering on blockchain hash continuity.

In many blockchain systems, transactions inside a block are organized using a Merkle tree. Each transaction is first hashed individually, and pairs of transaction hashes are repeatedly combined and hashed until a single root hash, called the Merkle root, is produced. The Merkle root summarizes all transactions in the block and allows efficient verification of transaction inclusion without requiring a node to download the entire block data. This structure supports lightweight verification, improves auditability, and reduces verification overhead in blockchain networks [1, 2].

Figure 3 illustrates how individual transaction hashes are combined pairwise to generate intermediate hashes and finally the Merkle root stored in the block header.

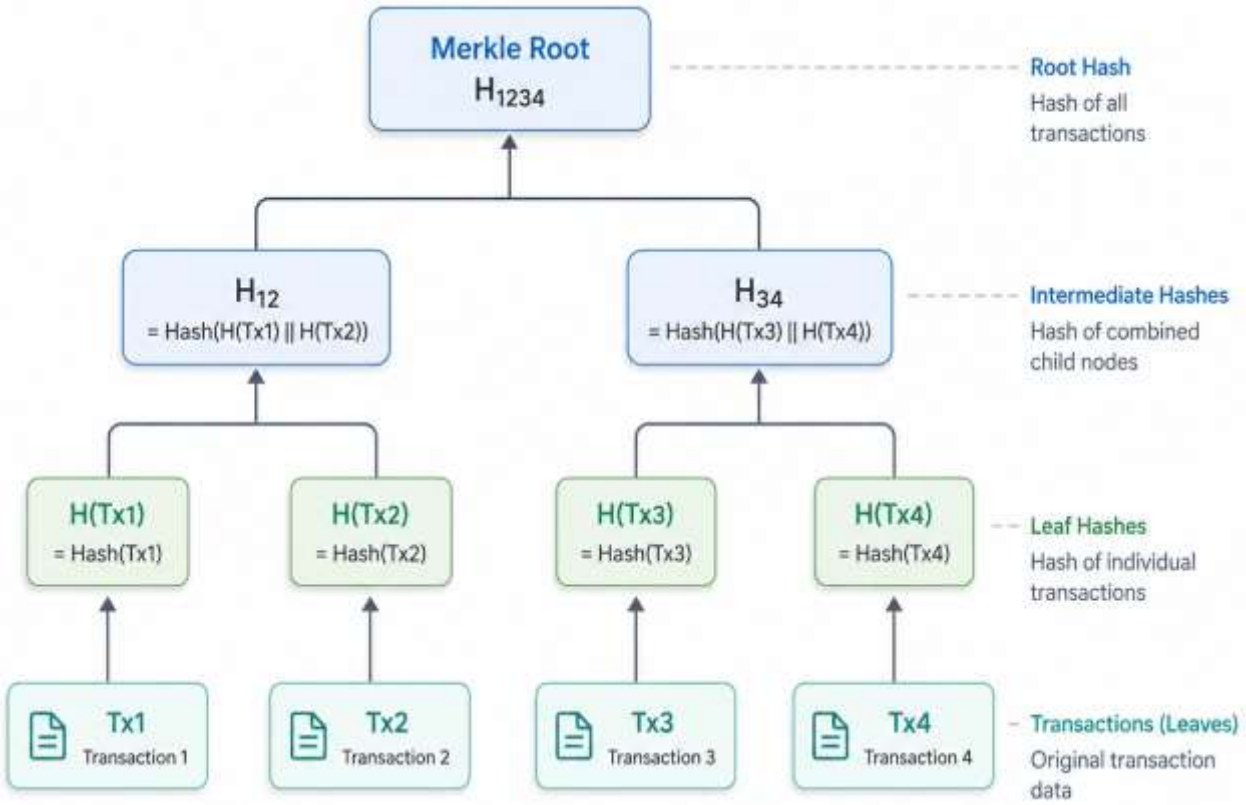

Figure 3. Merkle tree structure for summarizing transactions into a single Merkle root.

2.2 Transaction Lifecycle

The transaction lifecycle begins when a user creates and digitally signs a transaction using cryptographic keys. The transaction is then broadcast to the peer-to-peer network, where nodes check validity conditions such as signature correctness, balance availability, and protocol compliance [1–3]. Valid transactions are placed into a transaction pool until they are selected for inclusion in a block. After a candidate block is formed, the consensus mechanism determines whether the block becomes part of the ledger. Once accepted, the block is propagated across the network, and nodes update their local ledger state. This process provides a shared historical record of transactions, but its efficiency depends on block size, propagation delay, validator behavior,

and consensus overhead [2, 3, 10].

Figure 4 summarizes the transaction lifecycle from transaction creation to permanent block addition.

Figure 4. Transaction lifecycle in a blockchain network.

Table 2. Blockchain network types and architectural characteristics.

Type	Access	Control	Consensus
Public	Open	Decentralized	PoW, PoS
Private	Restricted	Single organization	PoA, PBFT
Consortium	Permissioned	Multiple organizations	PBFT, PoA
Hybrid	Mixed	Shared control	PoS, PoA, custom

3. PEER-TO-PEER NETWORKING AND NODE ROLES

Blockchain networks rely on peer-to-peer (P2P) communication rather than centralized coordination. In this model, nodes directly exchange transactions, blocks, and state information. P2P networking improves resilience because the system does not depend on a single server; however, it also introduces challenges such as propagation delay, network partitioning, malicious peers, and bandwidth overhead [1–3].

Different nodes may perform different functions. Full nodes store and validate the complete ledger, thereby contributing to decentralization and auditability. Light nodes store only partial information and depend on full nodes for verification, which improves usability for resource-constrained devices. Miner or validator nodes participate in block creation according to the consensus protocol. In permissioned frameworks, ordering nodes or validator committees may be responsible for transaction ordering and finality [5, 6, 12].

The distribution of node responsibilities directly affects security and performance. A highly decentralized network improves censorship resistance, but it may increase latency and consensus complexity. In contrast, permissioned systems with known validators can achieve higher throughput and faster finality, but they rely on stronger trust assumptions. Therefore, node architecture must be selected according to the intended deployment context.

Blockchain network type strongly influences architectural design, consensus selection, and governance. Public blockchains prioritize openness and decentralization, while private and consortium blockchains generally emphasize controlled participation, faster finality, and institutional governance. Therefore, network type should be considered before selecting a consensus mechanism or scalability strategy.

4. CONSENSUS MECHANISMS

Consensus mechanisms allow distributed nodes to agree on a single valid ledger state. Without consensus, different nodes could maintain conflicting transaction histories, enabling double spending or inconsistent records. A consensus protocol must therefore provide agreement, validity, and fault tolerance while remaining practical under real network conditions [1–3].

Consensus design strongly influences blockchain scalability, energy consumption, finality, and decentralization. Public blockchains generally prioritize open participation and adversarial resistance, while private and consortium blockchains often

prioritize low latency and controlled access. The following subsections summarize four major consensus approaches.

Consensus mechanisms are central to blockchain architecture because they determine how nodes agree on ledger state, handle adversarial behavior, and balance decentralization with performance [4].

4.1 PoW: Proof of Work

Proof of Work (PoW) requires miners to solve computationally intensive puzzles before proposing a block. This mechanism makes block creation costly and helps protect the network against Sybil attacks and ledger rewriting [1–3]. PoW was essential in demonstrating decentralized consensus in Bitcoin, but it also introduces high energy consumption, probabilistic finality, and limited throughput [1, 10, 11, 13].

PoW is suitable for open networks where participants are unknown and adversarial assumptions are strong. However, its performance limitations make it less suitable for high-throughput enterprise systems, IoT platforms, and latency-sensitive applications. Research therefore explores alternatives that reduce resource consumption while maintaining acceptable security guarantees.

4.2 PoS: Proof of Stake

Proof of Stake (PoS) replaces computational competition with validator selection based on economic stake. Validators are chosen to propose or attest blocks according to protocol-specific rules, and dishonest behavior may be penalized through stake loss. Compared with PoW, PoS can reduce energy consumption and improve scalability while preserving economic incentives for honest participation [2, 3, 11].

The major challenge in PoS is designing secure incentive models that prevent centralization and discourage malicious validator behavior. Stake concentration, validator collusion, and governance influence remain important concerns. Nevertheless, PoS has become a major direction in modern blockchain protocol evolution because it supports more sustainable consensus.

4.3 PBFT: Practical Byzantine Fault Tolerance

Practical Byzantine Fault Tolerance (PBFT) is designed for environments where participants are known or permissioned. PBFT-based systems can achieve fast finality and high throughput because consensus is reached through message exchange among a defined set of nodes [2, 5, 6]. This makes PBFT suitable for enterprise and consortium blockchain systems.

The limitation of PBFT is communication overhead. As the number of participating nodes increases, the number of messages exchanged during consensus grows significantly. Therefore, PBFT is effective in small or medium permissioned networks but less suitable for large open networks [5, 12]. Performance studies on private blockchain platforms show that consensus choice significantly affects latency and throughput [5, 12].

4.4 PoA: Proof of Authority

Proof of Authority (PoA) relies on a limited set of approved validators whose identities are known. Instead of computational power or stake, authority and reputation determine who can validate blocks. PoA provides low latency, high throughput, and reduced resource consumption, making it useful for private, consortium, and enterprise blockchains [5, 6].

The main trade-off is reduced decentralization. Since validation power is concentrated among approved authorities, PoA systems depend on institutional trust and governance mechanisms. For applications requiring controlled participation, this trade-off may be acceptable; for fully open decentralized environments, it may be unsuitable.

4.5 Comparative Performance Analysis

Consensus mechanisms differ in their assumptions and performance characteristics. Table 3 summarizes the major differences among PoW, PoS, PBFT, and PoA.

Table 3. Comparison of major blockchain consensus mechanisms.

Consensus	Network Type	Energy Use	Finality	Strength	Main Limitation
PoW	Public/open	High	Probabilistic	Strong adversarial resistance	Low throughput and high energy cost
PoS	Public or hybrid	Low to medium	Faster than PoW	Energy efficiency and economic incentives	Stake concentration and governance risks
PBFT	Permissioned	Low	Immediate or near immediate	Low latency and high throughput	Communication overhead with many nodes
PoA	Private/consortium	Very Low	Fast	Efficient enterprise operation	Reduced decentralization and validator trust

5. BLOCKCHAIN FORKING

Forking refers to divergence in the blockchain protocol, codebase, or ledger history. Forks are important because they allow blockchain systems to evolve, fix vulnerabilities, introduce upgrades, and respond to governance disagreements [2, 3, 7]. However, forks may also create uncertainty, compatibility issues, and community fragmentation.

5.1 Codebase vs Live Forks

A codebase fork occurs when developers copy and modify existing blockchain software to create a separate project. Such forks may reuse architectural ideas but usually begin with independent rules, communities, or ledgers. In contrast, a live blockchain fork occurs within an operational network when nodes temporarily or permanently diverge on ledger history or protocol rules.

Live forks may be accidental or intentional. Accidental forks can occur due to network latency when two valid blocks are produced nearly simultaneously. These are usually resolved when one branch becomes dominant under the consensus rule. Intentional forks are planned protocol changes and may be used to introduce upgrades, repair vulnerabilities, or alter governance rules.

5.2 Soft Forks vs Hard Forks

Soft forks introduce backward-compatible rule changes. Nodes that do not upgrade may still recognize the new chain as valid if the new rules are stricter than the old rules. Soft forks are therefore less disruptive but may still require broad community and validator support.

Hard forks introduce changes that are not backward compatible. If all participants do not upgrade, the network may split into two incompatible chains. Hard forks can enable major protocol improvements, but they also create governance risks and may divide communities [2, 3, 7].

Forking also has governance implications because protocol upgrades require coordination among developers, validators, miners, users, and application providers. In decentralized networks, disagreement over technical changes may lead to persistent chain

splits, reduced community trust, or duplicated assets. In permissioned networks, fork management is usually more controlled, but governance policies must clearly define how upgrades, validator changes, and emergency patches are approved. Therefore, forking should be understood not only as a technical event but also as a governance and coordination mechanism.

Table 4 summarizes major fork types.

Table 4. Comparison of blockchain fork types.

Fork Type	Cause	Result
Accidental fork	Temporary network delay	Usually resolved automatically
Codebase fork	Software reuse and modification	New independent project
Soft fork	Backward-compatible upgrade	Same chain if accepted
Hard fork	Non-compatible protocol change	Possible permanent split

Figure 5 presents the major categories of blockchain forking mechanisms.

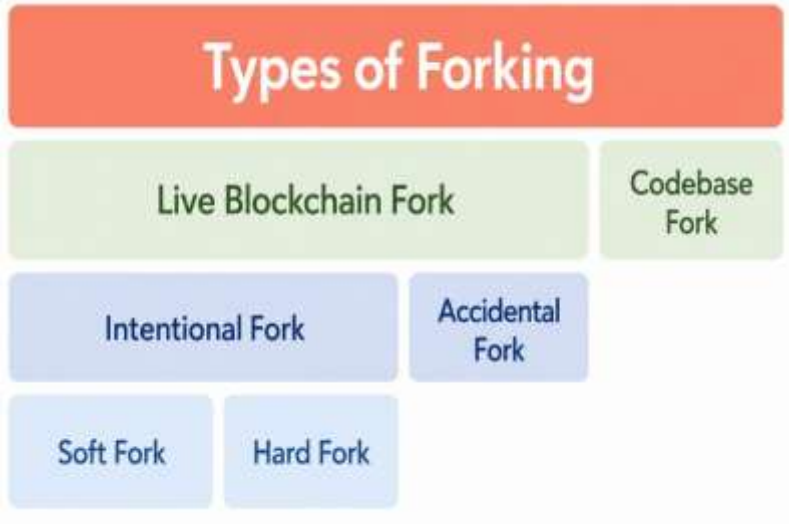

Figure 5. Classification of blockchain forking mechanisms.

6. SECURITY AND SCALABILITY CONSIDERATIONS

Blockchain security is supported by cryptographic hashing, digital signatures, replication, and consensus. However, blockchain systems are not immune to attacks. Major threats include double-spending, majority attacks, Sybil attacks, smart contract vulnerabilities, private key compromise, and network-level attacks [14–16]. Smart contract security is particularly important because programming errors may cause irreversible financial or operational losses [17, 18].

Security risks also differ across blockchain environments. Public blockchains face stronger adversarial conditions because participation is open and attackers may create multiple identities or attempt majority control. Permissioned blockchains reduce

some external attack risks by restricting participation, but they introduce different concerns such as validator collusion, insider misuse, access-control errors, and governance failure. This shows that blockchain security depends not only on cryptography but also on network membership, validator accountability, software correctness, and operational monitoring.

Scalability is another critical limitation. Public blockchains often face throughput constraints because all nodes may need to validate or store the same data. Increasing block size or reducing block interval may improve throughput, but it can also increase propagation delay and centralization risk. Storage growth also affects long-term node participation because maintaining full ledger history becomes increasingly expensive [3, 10, 11].

Scalability is further affected by the relationship between computation, communication, and storage. Increasing the number of transactions per block may improve throughput, but it can increase propagation delay and raise the probability of temporary forks. Similarly, requiring every full node to store and validate the complete ledger improves auditability but increases long-term storage cost. As a result, scalability solutions such as sharding, pruning, off-chain channels, and layer-2 protocols must be evaluated against their effects on decentralization, security, and data availability.

The blockchain trilemma describes the difficulty of simultaneously achieving decentralization, scalability, and security. Improvements in one dimension may weaken another. For example, permissioned systems may improve throughput but reduce decentralization, while highly decentralized public systems may sacrifice performance. Table 5 summarizes key issues.

Table 5. Security and scalability issues in blockchain systems.

Issue	Impact	Possible Direction
Double spending	Conflicting transactions	Strong finality rules
Majority attack	Ledger manipulation	Decentralized participation
Contract bugs	Irreversible errors	Formal verification
Low throughput	Limited processing capacity	Sharding and layer 2 systems
Storage growth	Costly full-node operation	Pruning and off chain storage
Network latency	Propagation delay and forks	Efficient propagation protocols

7. DISCUSSION

The analysis of architecture, consensus, forking, security, and scalability indicates that blockchain systems cannot be evaluated using a single performance metric. A system with high throughput may sacrifice decentralization, while a highly decentralized system may experience lower transaction speed and higher resource consumption. Similarly, a permissioned architecture may provide faster finality and easier governance, but it depends on trust in selected validators.

While application-oriented blockchain studies emphasize domain-level adoption and industrial value, architecture-oriented analysis is necessary to understand how consensus, network design, forking, and scalability constraints influence long-term protocol evolution [4, 19].

Another important observation is that architectural decisions are interdependent. Consensus mechanisms affect transaction finality, fork probability, energy consumption, and validator participation. Network design influences propagation delay, node synchronization, and resilience against attacks. Data-layer choices affect storage growth, auditability, and lightweight verification. Therefore, blockchain design requires a balanced evaluation of multiple layers rather than isolated optimization of one component. The paper also shows that protocol evolution is both technical and social. Forks, upgrades, validator rules, and governance models determine how blockchain systems adapt over time. Without clear upgrade mechanisms and governance processes, technical improvements may create community disagreement or compatibility risks. Thus, future blockchain systems must combine protocol

efficiency with transparent governance and long-term maintainability.

8. OPEN CHALLENGES IN PROTOCOL EVOLUTION

Future blockchain protocols must address several unresolved challenges. First, scalable consensus remains a central research problem. High-throughput systems must preserve security and decentralization while reducing latency and resource consumption. Second, interoperability is necessary because blockchain ecosystems are increasingly fragmented across platforms, standards, and governance models [3, 6, 10, 11].

Third, governance mechanisms require further development. Protocol upgrades, validator selection, dispute resolution, and fork management depend on governance processes that are often informal or inconsistent. Fourth, privacy-preserving architecture is essential for enterprise, healthcare, financial, and public sector systems. Techniques such as off-chain computation, zero-knowledge proofs, and permissioned access models may help balance privacy with auditability.

Fifth, formal verification and secure software engineering are required for smart contracts and protocol implementations [17,18]. Finally, energy-efficient and sustainable blockchain designs are necessary for long-term adoption. Future protocols should therefore combine technical scalability with security, governance clarity, and environmental responsibility.

Table 6 summarizes important future research directions for improving blockchain architecture and protocol evolution.

Table 6. Future research directions in blockchain architecture and protocol evolution.

Direction	Motivation	Expected Outcome
Scalable consensus	Latency and throughput limits.	Secure high-performance protocols.
Interoperability	Fragmented platforms.	Cross-chain communication.
Smart contract security	Irreversible coding errors.	Reliable decentralized applications.
Governance	Fork and upgrade coordination.	Stable protocol evolution.
Sustainability	Energy and storage costs.	Resource efficient systems.

9. CONCLUSION

This paper presented a focused survey of blockchain architecture and protocol evolution. It examined blockchain structure, transaction lifecycle, peer-to-peer networking, node roles, consensus mechanisms, forking models, and security-scalability trade-offs. The analysis shows that blockchain performance and reliability depend not only on cryptographic design but also on consensus choice, network structure, governance, and deployment context.

PoW, PoS, PBFT, and PoA each provide distinct benefits and limitations. Similarly, forks can support protocol evolution but may also introduce governance and compatibility risks. Future blockchain research should prioritize scalable consensus, interoperable architectures, privacy-preserving mechanisms, secure smart contract development, and sustainable protocol design. A technically grounded understanding of architecture and protocol evolution is essential for building reliable next-generation blockchain systems.

REFERENCES

[1] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system, 2008.

[2] Zheng, Z., Xie, S., Dai, H., Chen, X., and Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE international congress on big data (BigData congress)*, pages 557–564. IEEE, 2017.

[3] Zheng, Z., Xie, S., Dai, H.-N., Chen, X., and Wang, H. 2018. Blockchain challenges and opportunities: A survey. International journal of web and grid services, 14(4): 352–375.

[4] Hingu, M. and Pandey, K. Oct 2024. Blockchain Structural Dynamics: A Thorough Exploration of Architectural Configurations, Design Methodologies, Consensus Protocols, and Forking. Indian Journal of Natural Sciences, 15(86): 81638–81650.

[5] Hao, Y., Li, Y., Dong, X., Fang, L., and Chen, P. Performance analysis of consensus algorithm in private blockchain. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 280–285. IEEE, 2018.

[6] Polge, J., Robert, J., and Le Traon, Y. 2021. Permissioned blockchain frameworks in the industry: A comparison. ICT Express, 7(2): 229–233.

[7] Dabbagh, M., Sookhak, M., and Safa, N. S. 2019. The evolution of blockchain: A bibliometric study. IEEE Access, 7: 19212–19221.

[8] Syed, T. A., Alzahrani, A., Jan, S., Siddiqui, M. S., Nadeem, A., and Alghamdi, T. 2019. A comparative analysis of blockchain architecture and its applications: Problems and recommendations. IEEE Access, 7: 176838–176869.

[9] Dai, M., Zhang, S., Wang, H., and Jin, S. 2018. A low storage room requirement framework for distributed ledger in blockchain. IEEE Access, 6: 22970–22975.

[10] Xie, J., Yu, F. R., Huang, T., Xie, R., Liu, J., and Liu, Y. 2019. A survey on the scalability of blockchain systems. IEEE Network, 33(5): 166–173.

[11] Zhou, Q., Huang, H., Zheng, Z., and Bian, J. 2020. Solutions to scalability of blockchain: A survey. IEEE Access, 8: 16440–16455.

[12] Xu, X., Sun, G., Luo, L., Cao, H., Yu, H., and Vasilakos, A. V. 2021. Latency performance modeling and analysis for hyperledger fabric blockchain network. Information Processing & Management, 58(1): 102436.

[13] Kumar, G., Saha, R., Rai, M. K., Thomas, R., and Kim, T.-H. 2019. Proof-of-work consensus approach in blockchain technology for cloud and fog computing using maximization-factorization statistics. IEEE Internet of Things Journal, 6(4): 6835–6842.

[14] He, D., Choo, K.-K. R., Kumar, N., and Castiglione, A. 2018. IEEE Access special section editorial: Research challenges and opportunities in security and privacy of blockchain technologies. IEEE Access, 6: 72033–72036.

[15] Saad, M., Spaulding, J., Njilla, L., Kamhoua, C., Shetty, S., Nyang, D., and Mohaisen, D. 2020. Exploring the attack surface of blockchain: A comprehensive survey. IEEE Communications Surveys & Tutorials, 22(3): 1977–2008.

[16] Singh, S., Hosen, A. S., and Yoon, B. 2021. Blockchain security attacks, challenges, and solutions for the future distributed iot network. IEEE Access, 9: 13938–13959.

[17] Porru, S., Pinna, A., Marchesi, M., and Tonelli, R. Blockchain-oriented software engineering: challenges and new directions. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 169–171. IEEE, 2017.

[18] Liu, J. and Liu, Z. 2019. A survey on security verification of blockchain smart contracts. IEEE Access, 7: 77894–77904.

[19] Hingu, M. and Pandey, K. Aug 2024. Harnessing Blockchain Technology: A Comprehensive Review of Application Domains, Industry Transformations, and Future Prospects. Indian Journal of Natural Sciences, 15(85): 78227–78241.