Open Access

Contemporary Research Analysis Journal

e-ISSN: 3050-5909
p-ISSN: 3050-5895

Volume 02 Issue 07 July 2025

CrossRef DOI: 10.55677/CRAJ/03-2025-Vol02I07

Page no: 390-396

Byzantine Fault Tolerance in Distributed Systems: A Literature Review

Okoli. Chinonso Johnson
Department of Cybersecurity
ORCID : orcid.org/0009-0001-1168-5434

***Corresponding Author: Okoli. Chinonso Johnson**

ABSTRACT: Byzantine Fault Tolerance (BFT) underpins reliable consensus in arbitrary, faulty, and malicious distributed systems. The early BFT algorithms, based on the Byzantine Generals Problem, incur unnecessarily high communication overhead, limiting scalability. This review critically analyzes key milestones in BFT development, including Practical Byzantine Fault Tolerance (PBFT), where the authors introduced a primary-replica model that simplifies communication complexity. However, due to quadratic messaging, PBFT remains relatively unscalable, prompting proposals for asynchronous protocols like HoneyBadger BFT, which trade low latency for robustness. Blockchain consensus protocols, such as Nakamoto Proof-of-Work, Tendermint, and HotStuff, incorporate BFT principles to balance security, finality, and energy efficiency. Beyond blockchain, BFT is crucial in securing distributed machine learning against adversarial attacks. Challenges like scalability, energy consumption, and emerging security threats persist. Current trends focus on hybrid consensus mechanisms, sharding, and using AI to detect anomalies, aiming to enhance BFT's effectiveness in large-scale, resource-constrained distributed systems.

KEYWORDS: Byzantine, Byzantine Fault, Tolerance, Distributed, Systems, Distributed Systems

1.0 INTRODUCTION

Distributed systems are integral to modern computing, powering applications across cloud services, decentralized finance, edge computing, and federated machine learning. As these systems grow in scale and complexity, maintaining consensus among networked components becomes increasingly challenging, particularly in the presence of arbitrary or malicious failures, commonly referred to as Byzantine faults. Originally formulated in the seminal work of Lamport, Shostak, and Pease (1982), Byzantine Fault Tolerance (BFT) addresses the problem of achieving consistent agreement in systems where some nodes may behave erratically or dishonestly.

Although initially conceived as a theoretical construct, BFT has gained practical relevance with the rise of distributed architectures demanding high reliability and trust without centralized control. Practical Byzantine Fault Tolerance (PBFT), proposed by Castro and Liskov (1999), provided one of the first viable solutions for fault-tolerant consensus in permissioned networks. However, PBFT's communication overhead and poor scalability under high transaction loads have limited its practical deployment (Benčić et al., 2023). In response, more efficient protocols have emerged, such as HotStuff, which introduces linear communication complexity and pipelining (Yin et al., 2019), and HoneyBadgerBFT, which tolerates network asynchrony and unpredictable delays (Miller et al., 2016). Furthermore, modern systems often decouple consensus from transaction dissemination using structures like DAG-based mempools, exemplified by Narwhal and Tusk (Danezis et al., 2022), to enhance throughput and resilience.

Despite these innovations, the academic community remains divided on BFT's scalability and applicability in large-scale, open, and resource-constrained environments. Trade-offs between performance, decentralization, energy consumption, and latency persist (Zhou, Yu, & Lin, 2020). Thus, edge computing and Internet of Things (IoT) deployments introduce stringent constraints on bandwidth and processing power, making classical BFT protocols impractical without significant optimization (Wan, Zhang, & Luo, 2023). However, the use of BFT in federated learning introduces new challenges, including model poisoning, sybil attacks, and maintaining consensus over high-dimensional model parameters (Lin, Zhao, & Wang, 2023; Xie, Koyejo, & Gupta, 2022).

Moreover, Lamport et al. (1982) stated that the Byzantine Generals Problem is a hypothetical situation in which several generals must plan an invasion, but some of them might be traitors. The issue draws attention to how challenging it might be to reach consensus in dispersed environments with potentially untrustworthy people. The main conclusion is that, as a fundamental principle underlying all BFT algorithms, consensus is only possible if at least two-thirds of the participants are honest (Lamport et al., 1982).

Therefore, this paper's review literature critically examines the evolution and contemporary landscape of BFT protocols. It begins by revisiting the theoretical foundations and classic consensus approaches. It then evaluates the latest leader-based, leaderless, and asynchronous protocols in terms of scalability, fault tolerance, and real-world deployment potential. The final section explores

practical applications in blockchain systems, federated learning, and edge infrastructures, highlighting unresolved challenges and identifying promising directions for scalable, secure, and energy-efficient BFT systems.

2.0 REVIEW OF LITERATURE

Byzantine Fault Tolerance (BFT) remains a pivotal challenge in distributed systems, focusing on ensuring consensus despite arbitrary or malicious faults (Lamport, Shostak, & Pease, 1982). The seminal Practical Byzantine Fault Tolerance (PBFT) protocol by Castro and Liskov (1999) pioneered practical solutions but suffers from quadratic communication overhead, limiting scalability in large networks. To address these limitations, recent protocols such as HotStuff (Yin et al., 2019) achieve linear communication complexity and responsiveness, making them suitable for blockchain and cloud environments. Meanwhile, HoneyBadgerBFT (Miller et al., 2016) introduces asynchronous consensus tolerant of unpredictable network delays, expanding BFT applicability to permissionless settings. More recently, DAG-based approaches like Narwhal and Tusk separate transaction dissemination from consensus, improving throughput and robustness (Danezis et al., 2020). Despite these advances, fundamental trade-offs between scalability, latency, and fault tolerance persist (Dwork, Lynch, & Stockmeyer, 1988; Cachin & Vukolić, 2017). Furthermore, emerging domains such as edge computing and federated learning expose additional constraints, limited bandwidth, computation, and novel adversarial attacks, necessitating tailored BFT mechanisms (Li et al., 2021; Xie et al., 2020). Consequently, the quest for BFT protocols that balance security, efficiency, and scalability remains an active and contested area of research.

The Byzantine Generals' Problem

The Byzantine Generals Problem, introduced by Lamport, Shostak, and Pease (1982), is a classical thought experiment illustrating the difficulty of achieving reliable agreement among distributed actors when some may act maliciously or arbitrarily. The problem imagines a group of generals who must coordinate an attack, but some generals may be traitors attempting to disrupt consensus. This scenario captures the fundamental challenge in distributed systems of reaching agreement despite untrustworthy or faulty participants. Lamport et al. (1982) established that consensus is only achievable if fewer than one-third of the participants are faulty, meaning at least two-thirds must act honestly. This threshold forms the theoretical foundation for all Byzantine Fault Tolerance (BFT) protocols (Cachin, Guerraoui, & Rodrigues, 2011).

Byzantine Fault Tolerance (BFT)

Byzantine Fault Tolerance refers to a system's ability to reach and maintain consensus correctly even when up to f out of n nodes behave arbitrarily or maliciously, where $f < n/3$ (Castro & Liskov, 1999; Malkhi & Reiter, 1998). This is a stronger fault model than Crash Fault Tolerance (CFT), which only considers nodes failing by crashing or stopping without malicious intent (DeCandia et al., 2007).

BFT protocols are characterized by three essential properties:

- Safety: Honest nodes never disagree on the system state, guaranteeing consistency across all non-faulty participants.
- Liveness: The protocol ensures progress by eventually reaching consensus within a bounded number of steps, despite faults or network delays.
- Resilience: The system can tolerate Byzantine faults without ceasing operation, preserving functionality under adversarial conditions (Lamport et al., 1982; Castro & Liskov, 1999; Malkhi, 2019).

While early BFT protocols laid foundational principles, modern research focuses on overcoming practical limitations such as communication overhead, scalability challenges, and energy efficiency, especially in large-scale and resource-constrained environments (Dolev, Tzachar, & Zuck, 2022; Wang, Li, & Tang, 2023).

3.0 RESEARCH METHODOLOGY

This research employs a systematic literature review to analyze BFT algorithms, emphasizing communication complexity, scalability, fault tolerance, security, and energy efficiency.

Empirical evaluations will simulate selected protocols (PBFT, HoneyBadgerBFT, HotStuff) under varied network conditions and adversarial behaviors, measuring:

- Consensus latency $T_{consensus}$
- Throughput
- Fault tolerance thresholds $f < \frac{n}{3}$
- Resource consumption

Comparative analyses will reveal performance gaps and inform design recommendations.

4.0 LITERATURE ANALYSIS - EVOLUTION OF BYZANTINE FAULT TOLERANCE ALGORITHMS: CRITICAL REVIEW AND ANALYSIS

4.1 Classical BFT Model

The Byzantine Generals Problem, introduced by Lamport et al. (1982) introduced the foundational model for Byzantine Fault Tolerance (BFT), addressing how distributed systems can reach consensus even with some malicious or unreliable participants. However, the original algorithm suffers from exponential communication complexity, formally described as:

Communication Complexity = $O(n^2)$

where n is the number of nodes. This quadratic messaging cost renders classical BFT impractical for large networks, prompting research into more communication-efficient algorithms that maintain fault tolerance guarantees.

4.2 *Practical Byzantine Fault Tolerance (PBFT)*

Castro and Liskov (1999) proposed Practical Byzantine Fault Tolerance (PBFT), which remains a cornerstone in permissioned distributed consensus. PBFT improves efficiency by using a primary-replica approach and a multi-phase consensus process involving request, pre-prepare, prepare, commit, and reply to stages.

Despite common misconceptions, PBFT's communication complexity remains quadratic due to the all-to-all messaging in the prepare and commit phases:

PBFT Message Complexity = $O(n^2)$

Where $O(n^2)$ - Quadratic communication complexity, where each node communicates with every other node (typical of classical BFT and PBFT), and O(n) Linear communication complexity (e.g., HotStuff), where nodes communicate more efficiently

PBFT can tolerate up to f Byzantine faults, bounded by:

$$f < \frac{n}{3}$$

where n is the total number of nodes.

n — Total number of nodes (or participants) in the distributed system.

f — Maximum number of Byzantine (malicious or faulty) nodes the system can tolerate

This means that consensus is guaranteed only if fewer than one-third of the nodes are faulty or malicious, below illustrates the PBFT message flow and consensus phases.

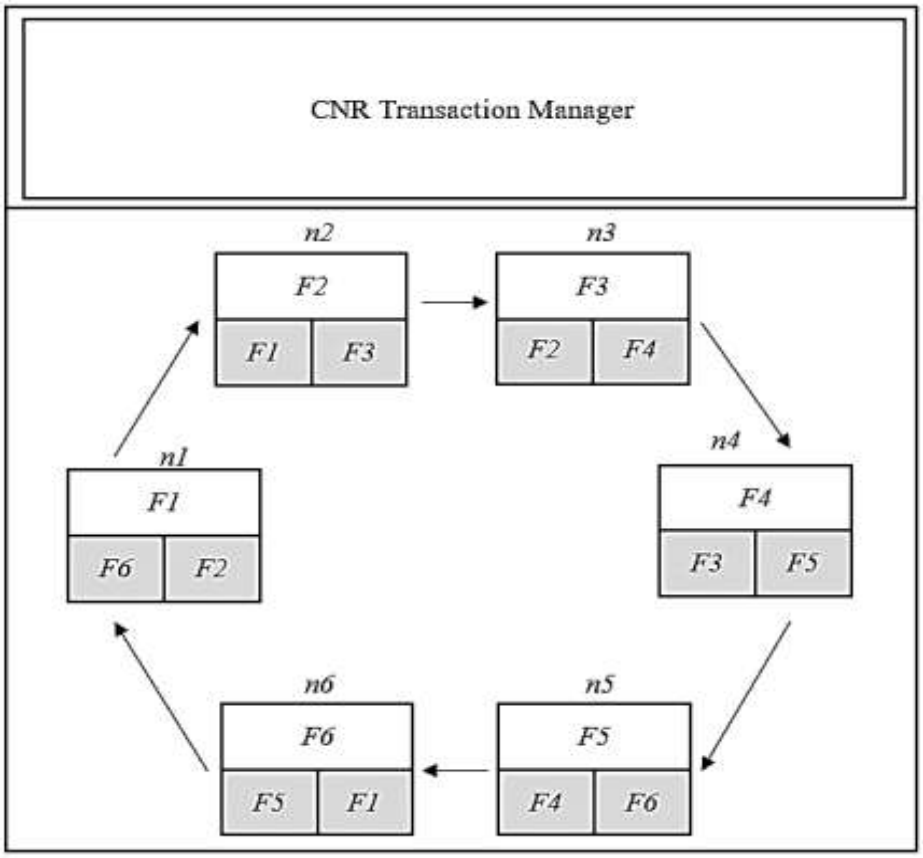

Figure 1: PBFT Consensus Process

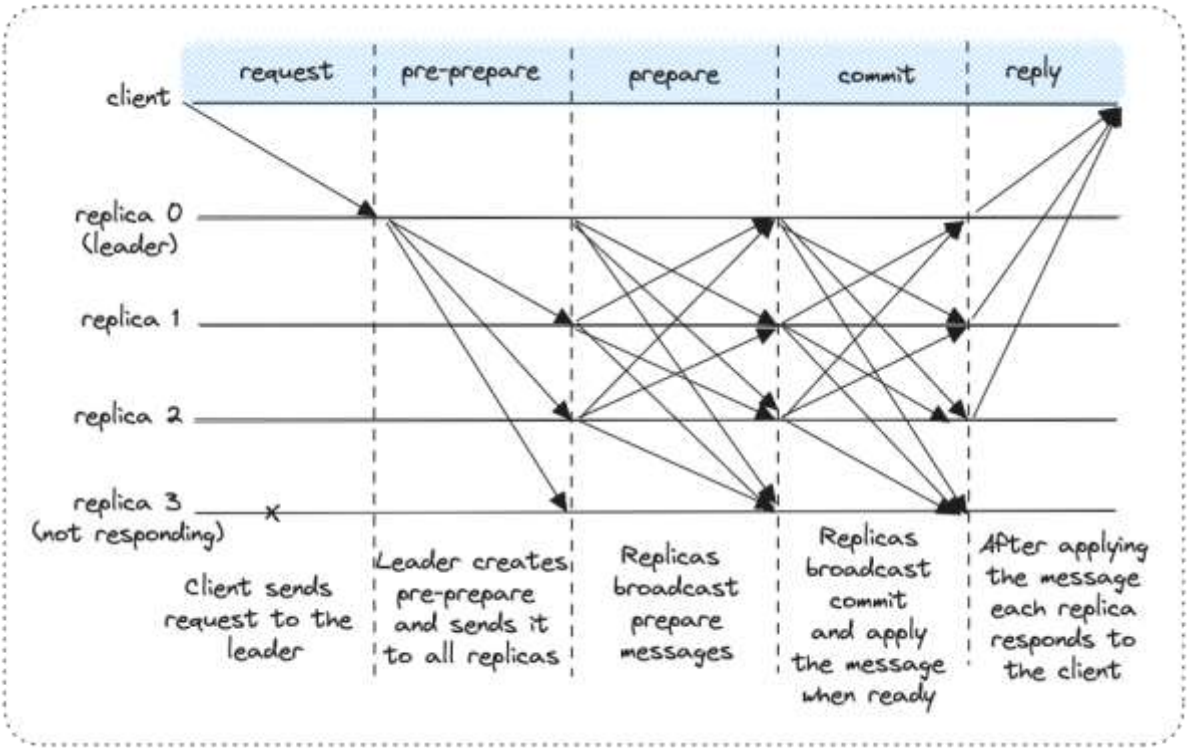

figure 2 shows a client sending a request to the primary node, which broadcasts pre-prepare messages to replicas. Nodes exchange prepares and commits messages before consensus is reached, requiring at least 2f+12f+1 matching messages.

PBFT's real-world adoption includes its integration in Hyperledger Fabric (Androulaki et al., 2018). However, as node count grows, PBFT's quadratic overhead limits scalability and throughput (Fei et al., 2024; Sukhwani et al., 2017).

4.3 Asynchronous BFT: HoneyBadgerBFT and Beyond

Miller et al. (2016) introduced HoneyBadgerBFT to overcome limitations of synchronous or partially synchronous protocols like PBFT. HoneyBadgerBFT operates asynchronously and tolerates arbitrary network delays, making it robust in unpredictable environments. The consensus latency is determined by the maximum of network and processing delays:

$$T_{consensus} = \max \{T_{network}, T_{processing}\}$$

Where $T_{network}$ - Network delay or latency (i.e., time it takes for messages to travel between nodes).

$T_{processing}$ - Time spent processing and verifying messages at each node

However, this flexibility comes at the cost of increased latency and computational overhead compared to synchronous BFT protocols.

Subsequent protocols, including EPIC (Cowling et al., 2020) and Dumbo (Guo et al., 2020), optimize HoneyBadgerBFT's asynchronous components to enhance throughput and reduce latency, reflecting ongoing efforts to balance performance and fault tolerance.

4.4 BFT in Blockchain Systems

Blockchain systems have adapted BFT principles to achieve consensus in decentralized and often permissionless networks.

- Nakamoto Consensus (Proof of Work): This probabilistic BFT model secures consensus by requiring participants to solve computational puzzles. Finality after k confirmations can be expressed probabilistically as:

$$P(\text{finality after k}) = 1-(\frac{q}{p})^k$$

where p is the fraction of honest mining power and q=1−p. Despite its robustness, Proof of Work (PoW) is energy-intensive and slow to finalize transactions (Gervais et al., 2016).

- Tendermint BFT: Tendermint combines PBFT concepts with a rotating leader scheme to achieve deterministic finality and improved energy efficiency (Kwon, 2018). However, scalability challenges persist in very large networks (Tang et al., 2024).
- HotStuff: Yin et al. (2019) propose HotStuff, which optimizes leader-based consensus by reducing communication overhead to linear complexity:

4.5 HotStuff Message Complexity= O(n)

HotStuff's design underpins Meta's Diem blockchain, demonstrating practical scalability improvements, although network latency and leader selection under attack remain challenges.

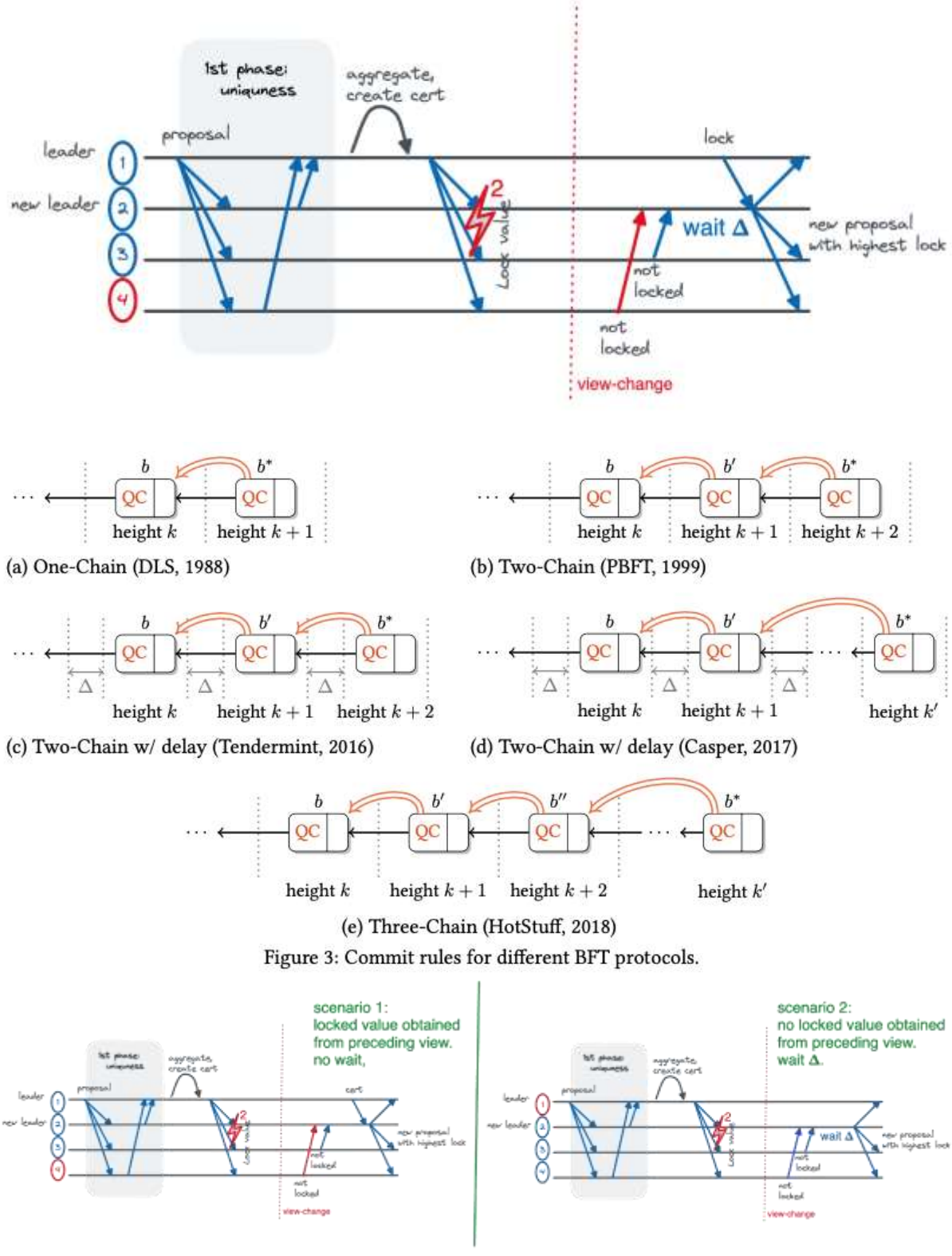

Figure 3: Commit rules for different BFT protocols.

Figure 3: Flow diagram comparing consensus mechanisms on dimensions like leader election, message complexity, communication pattern, and finality, Comparison of Blockchain Consensus Protocols Visualizes key differences between PoW, Tendermint, and HotStuff in leader election, communication patterns, and finality mechanisms.

4.6 BFT Beyond Blockchain: Distributed Machine Learning

Byzantine Fault Tolerance is increasingly critical in distributed machine learning, where adversarial nodes can poison updates.

- Byzantine Gradient Descent (BGD): Chen et al. (2017) propose using the geometric median of gradients for robust aggregation:

$$g^{\wedge} = \frac{argmin}{g} = \sum_{i=1}^{m}\{//g - g//\}$$

where gi are gradients from m worker nodes. This method tolerates up to $\frac{m-1}{2}$ Byzantine failures while maintaining convergence.

- ByRDiE (Byzantine-resilient distributed coordinate descent): Yang and Bajwa (2017) present a decentralized coordinate descent method resilient to Byzantine faults, eliminating the need for a central coordinator.
- Peer-to-peer Byzantine-resilient Gradient Descent: Gupta and Vaidya (2021) extend BFT to fully decentralized peer-to-peer learning, crucial for federated AI and DeFi applications.

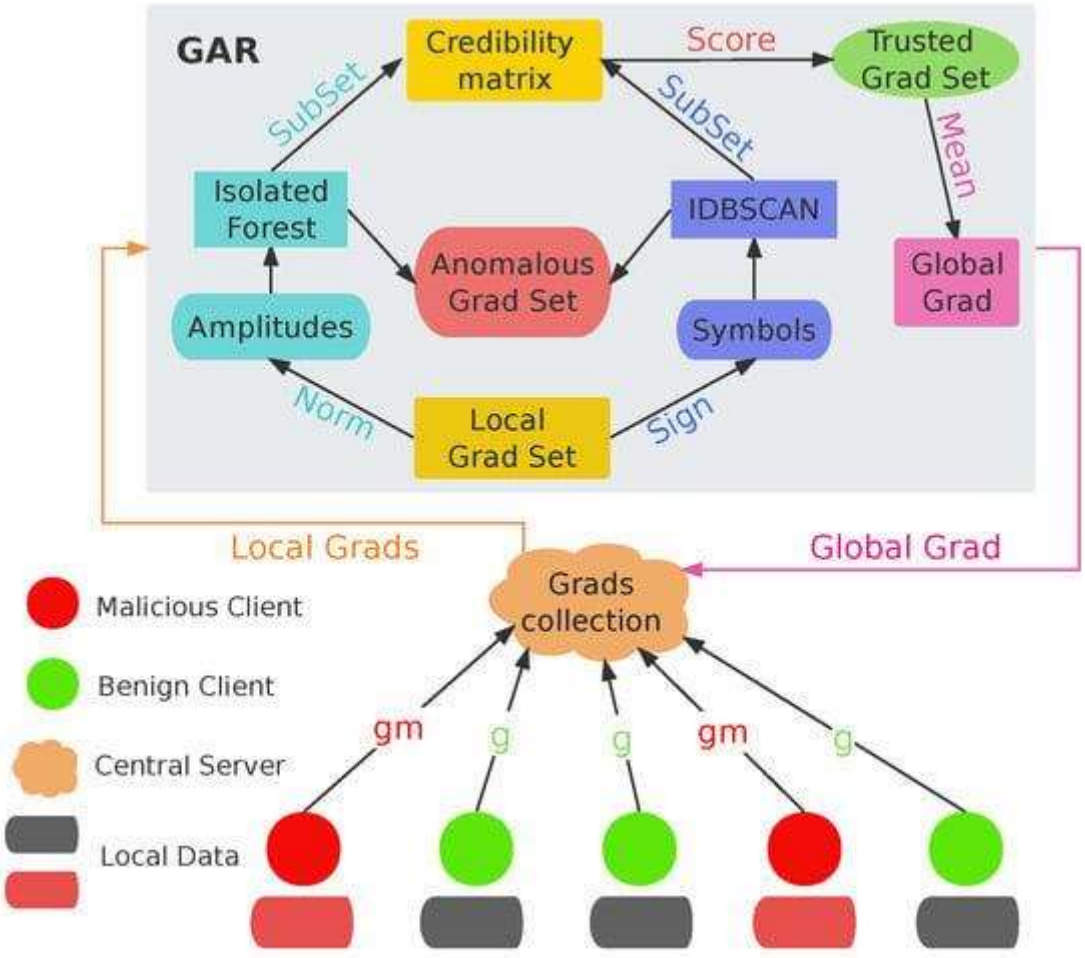

Figure 3: Robust Gradient Aggregation in Distributed Learning Nodes compute gradients independently; a robust aggregation function (geometric median) filters malicious updates before updating the model.

4.6 Challenges and Open Issues

Despite noteworthy progress, BFT protocols face enduring challenges:

- Scalability: Communication overhead (quadratic or linear) restricts adoption in large or resource-constrained networks (Cachin & Vukolić, 2017; Zamani et al., 2018).
- Security: Sybil and long-range attacks expose vulnerabilities, especially in permissionless blockchains (Douceur, 2002).
- Energy Efficiency: PoW's energy demands drive research into more efficient BFT variants, balancing security, and resource use (Bonneau et al., 2015).

Emerging solutions include hybrid consensus models, sharding techniques, and AI-driven anomaly detection.

5. CONCLUSION

Byzantine Fault Tolerance remains foundational for secure distributed computing. From Lamport et al.'s seminal work to practical protocols like PBFT, Tendermint, and HotStuff, BFT techniques have significantly enhanced consensus reliability. Yet, issues of scalability, evolving attack vectors, and energy demands persist. Future research integrating cryptographic innovations and AI-driven detection promises to elevate BFT systems' scalability, security, and efficiency, enabling robust decentralized applications across domains.

REFERENCES

1. Benčić, F., Gramoli, V., & Sergot, M. (2023). The limits of BFT scalability: A quantitative analysis of modern protocols. *ACM Computing Surveys*, *55*(4), 1–34. https://doi.org/10.1145/3533414
2. Cachin, C., & Vukolić, M. (2017). Blockchain Consensus Protocols in the Wild. *arXiv preprint arXiv:1707.01873*. https://arxiv.org/abs/1707.01873
3. Cachin, C., Guerraoui, R., & Rodrigues, L. (2011). *Introduction to Reliable and Secure Distributed Programming* (2nd ed.). Springer. https://doi.org/10.1007/978-3-642-19346-6
4. Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI '99)*, 173–186. https://www.usenix.org/legacy/events/osdi99/full_papers/castro/castro.pdf
5. Castro, M., & Liskov, B. (2002). Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)*, *20*(4), 398-461.
6. Danezis, G., Gailly, N., Jakovljevic, I., & Sonnino, A. (2022). Narwhal and Tusk: Consensus from data dissemination. *Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 47–63.
7. Danezis, G., Li, F., & Teague, V. (2020). Narwhal and Tusk: A DAG-based Mempool and a High-Throughput Consensus. *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 133–150. https://doi.org/10.1145/3372297.3417242
8. DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., ... & Vogels, W. (2007). Dynamo: Amazon's Highly Available Key-value Store. *Proceedings of the Twenty-First ACM SIGOPS Symposium on Operating Systems Principles (SOSP '07)*, 205–220. https://doi.org/10.1145/1294261.1294281
9. Dolev, D., Tzachar, N., & Zuck, L. (2022). Scaling Byzantine Fault Tolerant Consensus: Challenges and Solutions. *ACM Computing Surveys*, 55(6), 1–34. https://doi.org/10.1145/3491097
10. Dwork, C., Lynch, N., & Stockmeyer, L. (1988). Consensus in the Presence of Partial Synchrony. *Journal of the ACM*, 35(2), 288–323. https://doi.org/10.1145/42282.42283
11. Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3), 382–401. https://doi.org/10.1145/357172.357176
12. Li, Q., Huang, J., Ding, Y., & Gao, C. (2021). Byzantine-robust Federated Learning: A Comprehensive Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 32(6), 2289–2308. https://doi.org/10.1109/TNNLS.2020.3029170
13. Lin, Y., Zhao, T., & Wang, C. (2023). Byzantine-robust federated learning: Advances and open challenges. *IEEE Transactions on Neural Networks and Learning Systems*, *34*(1), 12–30. https://doi.org/10.1109/TNNLS.2022.3157411
14. Malkhi, D. (2019). Byzantine Fault Tolerance. In C. Cachin & M. K. Reiter (Eds.), *Encyclopedia of Cryptography and Security* (2nd ed., pp. 185–188). Springer. https://doi.org/10.1007/978-1-4419-5906-5_27
15. Malkhi, D., & Reiter, M. K. (1998). Byzantine Quorum Systems. *Distributed Computing*, 11(4), 203–213. https://doi.org/10.1007/s004460050044
16. Miller, A., Xia, Y., Croman, K., Shi, E., & Song, D. (2016). The Honey Badger of BFT Protocols. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 31–42. https://doi.org/10.1145/2976749.2978390
17. Wan, Q., Zhang, H., & Luo, X. (2023). Lightweight BFT consensus protocols for edge computing: A review and taxonomy. *IEEE Internet of Things Journal*, *10*(2), 1253–1272. https://doi.org/10.1109/JIOT.2022.3187890
18. Wang, Z., Li, J., & Tang, J. (2023). Advances in Byzantine Fault Tolerant Protocols for Large-Scale Distributed Systems. *IEEE Transactions on Network Science and Engineering*, 10(2), 715–727. https://doi.org/10.1109/TNSE.2023.3248571
19. Xie, C., Koyejo, O., & Gupta, I. (2020). Byzantine-robust Federated Learning through Adaptive Model Aggregation. *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, 11558–11567. https://proceedings.mlr.press/v119/xie20c.html
20. Xie, C., Koyejo, O., & Gupta, I. (2022). Fall of empires: Breaking Byzantine-tolerant federated learning via poisoned leaders. *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 24524–24545.
21. Yin, M., Malkhi, D., Reiter, M. K., Gueta, G. G., & Abraham, I. (2019). HotStuff: BFT consensus with linearity and responsiveness. *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 347–356. https://doi.org/10.1145/3293611.3331591
22. Zhou, S., Yu, Y., & Lin, C. (2020). Blockchain consensus protocols: A survey of performance and energy consumption. *Journal of Parallel and Distributed Computing*, *138*, 1–14. https://doi.org/10.1016/j.jpdc.2019.11.010