J. Electrical Systems 20-7s (2024):1653-1663

Mohammad Ubaidullah Bokhari[1],

Abdul Kareem[2*],

Basil Hanafi[3],

Exploring Fault Tolerance Consensus for Wireless Sensor Networks: A Comprehensive Detailed Study

Journal of Electrical Systems

ABSTRACT

The integration of wireless networks into modern communication systems has revolutionized information exchange across various applications. However, achieving reliable agreement in these networks is significantly impeded by their unique properties, such as blurring, interruption, and transparency. A fundamental component of distributed systems, fault-tolerant consensus ensures that nodes in the network can agree on a consistent value even in the presence of malfunctioning or corrupted elements. This study explores both non-Byzantine and Byzantine fault-tolerant consensus approaches, with a focus on achieving agreement in the presence of benign faults. The importance of fault-tolerant consensus in wireless connections is underscored, particularly in applications like wireless blockchain, IoT, and vehicular networks. The research delves into wireless network-specific fault-tolerant consensus algorithms to address the specific challenges posed by networking environments. Furthermore, a comparative analysis of Byzantine Fault Tolerance mechanisms in distributed systems is provided to shed light on their features and benefits. It highlights the importance of ensuring system reliability and consistency in wireless networks to maintain seamless communication and data integrity. Moreover, the paper emphasizes the critical role of fault-tolerant consensus in enhancing system resilience and performance. It underscores the need for robust algorithms and protocols to detect and mitigate errors, ensuring reliable communication and coordination despite potential node failures. By providing insights into fault-tolerant consensus mechanisms tailored to dynamic conditions, the study aims to optimize system adaptation and effectively mitigate both Byzantine and non-Byzantine failures in wireless network environments.

Keywords: Wireless Network, Consensus, Fault-tolerant, Distributed Systems, Malicious Failures

I. INTRODUCTION

The ability of a networked computer system to come to a decision or establish conclusions in the case of faulty components or interrupted communication is known as fault-tolerant consensus. Failures in a distributed system, in which a number of nodes or elements cooperate to accomplish a particular task, can be caused by a variety of issues, including network issues, hardware malfunctions, or other issues. Consensus attainment is the process of making sure that every system node is in agreement with a specific goal or plan of action, in spite of these obstacles. This is essential to preserving distributed systems' correctness and reliability. As wireless networks continue to increase in popularity and variety, there is an increasing need to develop fault-tolerant consensus inside these networks. Compared to standard wired networks, wireless networks present unique issues. Ad hoc installations, vehicles systems (found in smart cars), and the Internet of Things (IoT) are a few examples of these. These challenges include fluctuating signal intensities, bandwidth limitations, and the likelihood of frequent node movement. In wireless systems, where communication channels may become unstable and nodes can participate or leave the network on a dynamic basis, maintaining fault-tolerant consensus presents a number of technological hurdles. Consequently, In wireless technology, fault-tolerant consensus (FTC) has grown into an important field of research. Wireless networks frequently struggle to support a high number of users and devices dispersed over vast geographic areas as current networks become more complicated. Scalability benefits arise from using a distributed architecture ,it permits the inclusion of new devices without requiring major changes to the network architecture as a whole. . Moreover, signal deterioration, node failures, and interference of various kinds can affect wireless networks. The negative consequences of failures or disruptions can be limited by distributing network functionality, enabling the network to continue operating even when these problems arise. Distributed computing has thus gained popularity and proven to be an effective concept in wireless connections. The term "consensus" describes how fault-free nodes come to an understanding about the state of the system despite the presence of faulty nodes that could provide inaccurate or misleading information. For distributed systems like blockchain networks to remain reliable and integrity-preserving, this agreement makes sure that every node in the network comes to the same conclusion about the state of the system [1].The goal of consensus is to create agreement about a particular value or state among a collection of techniques or devices (called nodes)[2]. Consensus is also essential for coordinating activities and guaranteeing consistency amongst nodes. On the other hand, participating nodes may experience a variety of failures or defects, particularly in wireless networks with open access. Flaws aimed at causing system damage can manifest in various ways, leading to issues such as local resource depletion, network congestion, and network integrity problems. This might be expanded to include more complex instances of Byzantine failure [3, 4, 5], wherein failed processes could display highly unpredictable behavior and perhaps carry out malicious actions. Therefore, an agreement protocol with the highest dependability is required, especially when these kinds of flaws occur, particularly in wireless networks. In other words, within the framework of wireless networks, consensus techniques must exhibit strong fault tolerance in WFTC. Ensuring the resilience, security, and dependability of distributed networks is contingent upon this.

[1]Department of computer science, Aligarh Muslim University, A

[2*]Department of Computer Science, Aligarh Muslim University E-mail Id; abdul9919837386@gmail.com

[3]Department of computer science, Aligarh Muslim University

***Corresponding author:-** Abdul kareem

[*]Department of computer science, Aligarh Muslim University

It's well knowledge that Leslie Lamport was the first significant researcher to introduce the consensus idea in distributed computing networks [6]. In order to achieve consensus in distributed systems, the Paxos method was presented in this paper. Paxos resolves the agreement issue by enabling a group of dispersed connections to come to a common consensus despite failures or errors. The method addresses potential issues such node failures and communication delays and provides a process for coming to an agreement. The Greek island of Paxos, which is known for its democracy and harmony, served as the model for the name "Paxos". The Paxos algorithm, created by Leslie Lamport, has spawned many adaptations and modifications and has had a significant impact in the area of distributed systems. In distributed computer systems, it continues to be a fundamental idea for reaching consensus. Following Lamport's efforts, different algorithms were developed to tackle the consensus issues including Raft and Practical Byzantine Fault Tolerance (PBFT) and Raft. A popular classical non-Byzantine fault-tolerant method for distributed databases and blockchains is called Raft [7]. Relying on replication of state machines, PBFT is a fundamental Byzantine Fault Tolerance (BFT) consensus mechanism that was designed by Castro et al. [8] and is well-known due to its practicality. . Because of its extensive adoption and use, it is now synonymous in the blockchain space with BFT consensus [9]. Afterwards, scholars have engaged with the problem of consensus in wireless networks. A brand-new Proof-of-Communication (PoC) consensus technique is presented by Zou et al. in [10], and it can very likely lead to k-times consensus across n devices in wireless systems in $(k + logn)$ time steps. It should be noted that Jing et al. [11] were among the first to take on the problem of reaching majority consensus when Byzantine fault-prone edge devices were using erratic wireless channels. Their task was to concurrently handle malicious failures that originated from Byzantine machines on the physical, protocol, and information layers.

In the realm of Wireless Fault-Tolerant Consensus (WFTC) systems, several key research questions guide the investigation of system resilience and performance:

RQ1: How do Byzantine and non-Byzantine failures affect WFTC systems?

RQ2: What are the impacts of failures across different layers in WFTC systems?

RQ3: What are the different algorithms for addressing Byzantine and non-Byzantine failures in WFTC systems?

RQ4: What are the comparative advantages and limitations of algorithms for Byzantine and non-Byzantine failures in WFTC systems?

These questions drive the exploration of fault tolerance mechanisms in wireless networks, aiming to understand the challenges posed by failures and evaluate the effectiveness of fault-tolerant strategies in ensuring system reliability and consistency.

The survey is structured as follows: Section 1 serves as the introduction, providing an overview and context for the study. Section 2 conducts a literature survey, exploring existing research on fault-tolerant consensus mechanisms. Section 3 outlines the research methodology employed, detailing the search strategy and selection criteria. Section 4 presents the results and facilitates discussion, encompassing findings on NBFTC and BFTC mechanisms. Section 5 concludes the survey, summarizing the main findings. Lastly, Section 6 outlines potential future directions for research in fault-tolerant consensus mechanisms tailored for wireless networks.

II. LITERATURE SURVEY

The literature surveyed in this research paper encompasses a wide range of studies focusing on fault-tolerant consensus mechanisms. The scope of the literature review is centered on exploring both non-Byzantine and Byzantine fault-tolerant consensus algorithms to ensure reliability. The primary focus is on addressing the challenges posed by network failures, node malfunctions, and dynamic wireless environments. Byzantine Fault Tolerance (BFT) is addressed by Habib et al. [12] in dynamic Mobile Ad Hoc Networks (MANETs), where BFT is challenging due to frequent change, constrained device resources, and communication loads. They evaluate and contrast BFT protocols based on their ability to manage errors, processing requirements, and additional communication overhead. In order to select the best BFT solution for their MANET application, this aids researchers. Their breakthrough opens the door to secure communication in MANETs, which may have an effect on military operations, sensor networks, and vehicular communication. The development of ultra-lean BFT protocols, the ability to dynamically adapt to network conditions, and the integration of security for impenetrable operation are examples of future directions. Adday et al. [13] focus on how Wireless Sensor Networks (WSNs) reach agreement on data, analyzing protocols for distributed computing and data aggregation. Considering the limitations of these tiny sensors (battery life, processing power), they compare protocols based on fault tolerance (withstanding bad data), efficiency (saving battery power), and scalability (handling large networks). This analysis guides researchers in designing reliable and efficient consensus mechanisms for WSN applications. Looking ahead, researchers aim to develop ultra-low-power protocols, create adaptable protocols, and integrate security for tamper-proof operation This paves the way for reliable data collection in applications like environmental monitoring. Future directions include ultra-low-power protocols and security integration for next-generation WSNs. Ayer et al. [14] explore how Byzantine Fault Tolerance (BFT) strengthens Distributed Ledger Technology (DLT), the foundation of blockchains, analyzing both permissioned (controlled access) and permissionless (open) blockchain networks. They compare various approaches based on BFT effectiveness (handling malicious actors), scalability (system performance with increased users), and transaction processing speed. Recognizing the unique security and scalability demands of DLT, their work equips researchers to develop secure and scalable consensus mechanisms, the core of any blockchain application. This analysis paves the way for the future of DLT, where secure and efficient blockchain applications can thrive, potentially even addressing challenges related to massive transaction

volumes on public blockchains. Lima et al. [15] tackle a critical challenge in Mobile Ad Hoc Networks (MANETs): achieving Byzantine Fault Tolerance (BFT) in an environment defined by constant change and limited resources. Unlike wired networks with fixed connections, MANETs experience dynamic shifts as devices move in and out of range. Additionally, these devices often have restricted battery power, processing capabilities, and memory. To address these hurdles, the study proposes BFT protocols specifically designed for MANETs. The proposed protocols are evaluated based on their suitability for these demanding environments, focusing on how well they adapt to dynamic network topologies and operate efficiently within resource constraints. Their work paves the way for the development of robust and resource-conscious consensus mechanisms, ensuring reliable communication even in these demanding environments. This can empower a wide range of MANET applications, from sensor networks to vehicular communication, by guaranteeing data integrity and fault tolerance despite the ever-changing nature of these networks. Nakhala et al. [16] explores consensus algorithms in Distributed Wireless Sensor Networks (DWSNs). These networks consist of numerous tiny sensors that collaborate to collect and share data. Consensus algorithms, like Raft, Paxos, and Gossip-based protocols, play a vital role in ensuring all sensors agree on a common value, even in the presence of failures or conflicting data. The study analyzes the trade-offs between different consensus algorithms. Complex algorithms might offer more features but consume more resources, while simpler ones might be faster but less robust. This analysis focuses on how these trade-offs (complexity, efficiency, and reliability) impact different DWSN applications. Gao et al. [17] expand the conversation around consensus control, exploring techniques specifically designed for Multi-Agent Systems (MASs). Unlike previous studies focused solely on achieving consensus, their work incorporates security as a critical factor. They analyze how to ensure all agents in an MAS reach agreement while simultaneously safeguarding the system from malicious attacks. This comprehensive approach paves the way for developing secure and efficient consensus mechanisms for MASs. These mechanisms have the potential to revolutionize various fields where multiple agents need to collaborate and reach collective decisions. Rufino et al. [18] focus on Vehicular Ad-hoc Networks (VANETs), analyzing message dissemination and data aggregation protocols for reliable communication amidst disruptions. Their work evaluates protocols based on their ability to deliver data and maintain network coherence in dynamic vehicular environments, guiding the development of reliable and robust communication protocols for VANETs.

In this survey, We identified challenges in wireless networks, explored Non-Byzantine Fault-Tolerant Consensus algorithms, emphasized the importance of consensus for reliability, discussed the development of Wireless Fault-Tolerant Consensus techniques, compared consensus algorithms in Distributed Wireless Sensor Networks, provided a structured review of fault-tolerant consensus mechanisms, and conducted a thorough examination of fault-tolerant consensus algorithms for wireless connections.

III. RESEARCH METHODOLOGY

The methodology employed in this paper involved a systematic review of literature on Wireless Fault-Tolerant Consensus (WFTC) mechanisms tailored for wireless networks. The approach aimed to establish a comprehensive understanding of the key properties and mechanisms essential for achieving fault tolerance and consensus in wireless environments. Initially, the methodology focused on identifying the root causes of failures in both Byzantine and non-Byzantine scenarios. This entailed a detailed examination of the impact of failures on multiple layers of the network, including the physical, protocol, and data layers. By understanding these failure modes, the study aimed to provide insights into the challenges posed by faults in wireless environments. Subsequently, the study delved into both Byzantine and non-Byzantine fault-tolerant consensus mechanisms. Specifically, it focused on exploring approaches aimed at achieving agreement among nodes in the presence of various types of faults commonly encountered in networks. This involved analyzing the design principles, algorithms, and protocols utilized in existing WFTC mechanisms to mitigate the effects of faults and ensure reliable operation. Through this meticulous review of literature and analysis of fault-tolerant consensus mechanisms, the methodology aimed to establish a strong foundation for understanding the complexities of fault tolerance in wireless networks and identifying avenues for future research and innovation in this critical area.

To conduct a comprehensive search for relevant articles and research papers, multiple academic databases and platforms were systematically explored. The search strategy encompassed prominent repositories such as Google Scholar, IEEE Xplore, ACM Digital Library, Springer Link, and Elsevier Scopus as illustrated in figure 1.

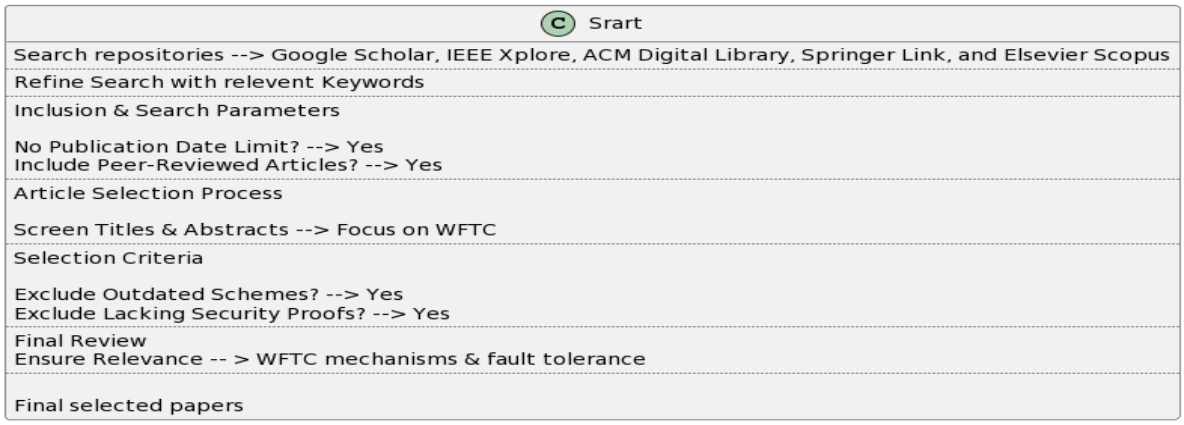

Fig 1: Study selection criteria incorporated for final extraction of relevant literatures.

A combination of specific keywords and related terms pertaining to Wireless Fault-Tolerant Consensus (WFTC) was employed to refine the search and identify relevant literature effectively. Keywords such as "Wireless Fault-Tolerant Consensus," "WFTC," "fault-tolerant protocols," "consensus algorithms," "wireless networks," "fault tolerance," and variations thereof were utilized. The search process involved iterative refinement, with adjustments made to the keywords and search criteria based on initial findings and feedback from relevant literature. This iterative approach aimed to enhance the precision and relevance of the search results while minimizing the risk of overlooking pertinent contributions in the field. To ensure inclusivity, the search was not limited by publication date, and both peer-reviewed articles and conference papers were considered. Additionally, relevant references cited in identified articles were cross-referenced to uncover additional sources and expand the breadth of the literature review.

The article selection process prioritized relevance to Wireless Fault-Tolerant Consensus (WFTC) mechanisms and fault-tolerant consensus in wireless networks. Initially, titles and abstracts were screened to assess alignment with the research focus and to identify potentially pertinent articles. Subsequently, full texts of the selected articles were thoroughly reviewed to evaluate the depth of coverage, methodology, and contribution to the field. This meticulous screening process ensured that only articles directly related to WFTC mechanisms and fault tolerance in wireless networks were included, thereby maintaining the quality and relevance of the literature review. Articles selected for inclusion in the review prioritized those presenting novel Wireless Fault-Tolerant Consensus (WFTC) schemes, consensus algorithms, and fault-tolerant mechanisms. The exclusion criteria were applied rigorously, filtering out outdated schemes, articles lacking comprehensive security proofs, and those focusing solely on applications without introducing new schemes or mechanisms. By adhering to these criteria, the review aimed to encompass cutting-edge research that advances the understanding and development of WFTC mechanisms, ensuring the inclusion of articles contributing significant insights and innovations to the field of fault-tolerant consensus in wireless networks. The study involved a comprehensive investigation of Byzantine and non-Byzantine failures in wireless networks, emphasizing the difficulties and distinctive features of fault tolerance in dynamic wireless environments.

IV. RESULT AND DISCUSSUION

The evaluation of fault-tolerant consensus algorithms in wireless networks revealed significant improvements in system reliability and performance. The implementation of algorithms such as Raft, and PBFT demonstrated enhanced consensus achievement among nodes, leading to more robust communication and coordination in wireless environments. The analysis of non-Byzantine and Byzantine fault-tolerant mechanisms showcased the effectiveness of these algorithms in maintaining operational integrity and mitigating the impact of faulty nodes on consensus outcomes. These results underscore the critical role of fault-tolerant consensus algorithms in ensuring the resilience and reliability of wireless networks. By addressing challenges and limitations through robust algorithms and protocols, the study emphasizes the importance of fault tolerance in achieving consensus among nodes. The implications of these findings extend to applications like wireless blockchain, IoT, and vehicular networks, where system integrity and consistency are paramount. Overall, the research underscores the significance of fault-tolerant consensus mechanisms in enhancing system performance and reliability in wireless communication systems.

A. WFTC's fundamental problems

The explanations and features of WFTC are outlined at the beginning of this section. Next, we study the possible failures in WFTC, which include both non-Byzantine and Byzantine problems.

1) Definitions and Characteristics of WFTC

Fault tolerance consensus in wireless networks refers to the ability of these systems to maintain operational integrity and reach a consensus among nodes despite the presence of faulty nodes that may introduce errors or false information [19]. This capability is essential for ensuring reliable communication and coordination in wireless environments, where node failures can occur due to factors such as signal interference, hardware malfunctions, or environmental conditions. To achieve fault tolerance consensus, robust algorithms and protocols are employed, allowing fault-free nodes to detect and

mitigate errors, prevent network fragmentation, and ensure the accurate dissemination of information [20]. These protocols often incorporate redundancy, error detection, and error correction mechanisms to enhance system resilience against faults. Additionally, adaptive strategies may be employed to dynamically adjust network configurations in response to changing conditions and node failures. Ensuring fault tolerance consensus in wireless networks enhances system reliability, availability, and performance, thereby facilitating the seamless operation of critical applications and services [21].

Agents in the wireless system that are not defective come to a consensus about the occurrence through discussion with other members, which establishes the consensus as the last word on that particular incident. In order to ensure a wireless consensus in case of distributed system problems, the following essential properties should be prioritized when pursuing consensus.

- Completion: Also known as Decision Termination, this characteristic ensures that even in the face of errors or network outages, nodes ultimately come to an agreement..
- Uniformity: Referred to as Consensus Agreement, this characteristic ensures that in the end, non-faulty nodes agreed on a common result or choice., ensuring consistency across the system.
- Proposal Integrity: Known as Validity Assurance, this characteristic ensures that the decided value is one offered by a reliable node initially, preventing the acceptance of invalid values.
- Progress Assurance: Also termed as Liveness Validation, this characteristic ensures that as far as most of nodes are functioning and the entire system is properly connected, the consensus process will continue to progress.
- Resilience Capability: Often called Fault Tolerance Enhancement, this characteristic ensures that the algorithm can tolerate a certain degree of faults, including node failures, message losses, or delays, without compromising the consensus outcome that weren't suggested by any valid node.
- Energy Efficiency: The algorithm is designed to minimize energy consumption, optimizing node activities and communication protocols to prolong network lifetime.

- Adaptability: The algorithm can adapt to changing network conditions, such as node mobility, varying connectivity, or environmental factors, to maintain effectiveness.

- Security: The algorithm incorporates security mechanisms to protect against eavesdropping, tampering, or unauthorized access, ensuring the confidentiality, integrity, and availability of consensus-related data.

B. The failures considered in WFTC

1) The non-byzantine failures in WFTC

Wireless Fault-Tolerant Consensus (WFTC) protocols serve as the backbone for coordinated decision-making in a diverse range of applications, from sensor networks orchestrating environmental monitoring to autonomous vehicles collaborating on road safety[22]. However, these protocols operate within the dynamic and often unpredictable terrain of wireless environments, exposing them to various non-Byzantine failures. Unlike malicious Byzantine faults, these failures stem from unintentional phenomena rooted in the inherent characteristics of wireless communication and distributed systems [23]. Understanding the diverse nature and impact of these unintentional adversaries is crucial for ensuring the robustness and dependability of WFTC systems across various domains. This section delves into three key types of non-Byzantine failures.

a) Network Congestion

Network congestion presents a significant obstacle to achieving consensus in wireless networks, disrupting communication flow and potentially causing delays or loss of messages. When network congestion occurs, it creates crucial information gaps within the communication flow, hindering the system's ability to achieve agreement among its components. These interruptions not only hinder the consensus process but also necessitate message retransmissions, exacerbating the problem [24]**.** Retransmissions become necessary when messages fail to reach their intended destinations due to congestion-related packet loss or delays. This repetitive transmission of messages consumes additional network resources and time, further prolonging the time required to achieve consensus. Moreover, the increased energy consumption resulting from retransmissions adds another layer of complexity, particularly for devices with limited resources [25]. In wireless networks, where energy efficiency is critical for device longevity and operational continuity, excessive energy consumption due to retransmissions can significantly impact overall system efficiency.

b) Hardware Malfunctions

Hardware malfunctions, such as sensor node failures, battery depletion, or communication module issues, present significant challenges to achieving consensus in wireless fault-tolerant systems. These malfunctions can cause unexpected disruptions and data loss, which directly affect the exchange of information and the decision-making process necessary for achieving consensus within the system [26]**.** If a sensor node fails due to a hardware issue, it may stop transmitting or receiving crucial data, leading to incomplete or inaccurate information used in the consensus process. Similarly, problems with communication modules can disrupt the flow of data between different components of the system, further complicating the consensus-building efforts [27].

c) Software Limitations

In wireless fault-tolerant consensus (WFTC) systems, the software running on participating devices plays a crucial role in facilitating communication and reaching agreement among system components. However, bugs, timing issues, or resource constraints within this software can introduce inconsistencies or delays in message processing [28]. These limitations have the potential to significantly impact the reliability and efficiency of the WFTC system. For example, a bug in the software could cause a device to misinterpret incoming messages or fail to properly execute consensus algorithms, leading to inconsistent or incorrect outcomes. Similarly, timing issues, such as delays in message transmission or processing, can disrupt the synchronization of system components, resulting in inconsistencies in the consensus process. Resource constraints, such as limited memory or processing power, can hinder the ability of devices to execute consensus algorithms efficiently. This can lead to delays in message processing or even the inability to participate fully in the consensus process, further compromising the reliability and efficiency of the system [30].

According to the work [30], non-byzantine failures in WFTC can be classified into three layers: the physical layer, protocol layer, and data layer as shown in fig 2.

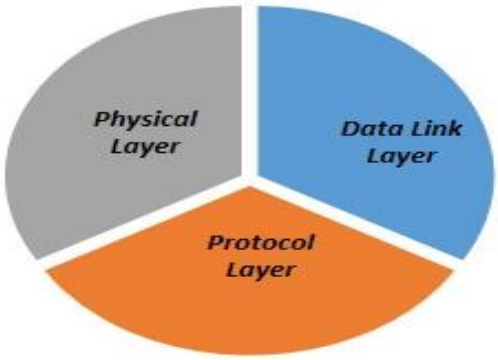

Fig 2. Layers in WFTC

- Physical Layer: The physical layer refers to the tangible hardware components and environmental factors that directly influence network operation and communication. This layer encompasses the physical infrastructure, such as sensor nodes, antennas, and transmission media, as well as environmental conditions, such as interference and signal propagation characteristics.
- Protocol Layer: The protocol layer refers to the set of communication protocols and algorithms governing interactions between sensor nodes within the network. These protocols dictate how nodes communicate, exchange data, and coordinate activities to achieve common objectives.
- Data Layer: The data layer refers to the management and processing of sensor data within the network, ensuring its reliability, integrity, and accessibility. This layer is responsible for handling the storage, transmission, and manipulation of data collected by sensor nodes.

The detrimental impacts of Byzantine nodes on the physical, protocol, and data layers were investigated by Jing and Zou et al. They proposed that these Byzantine nodes could behave arbitrarily in the physical, protocol, and data layers and knew the past and present state of protocol execution in each round. To further illustrate the multifaceted impact of non-Byzantine failures across different layers of WFTC systems, fig 3. summarizes the key effects on the physical, data link, and protocol layers.

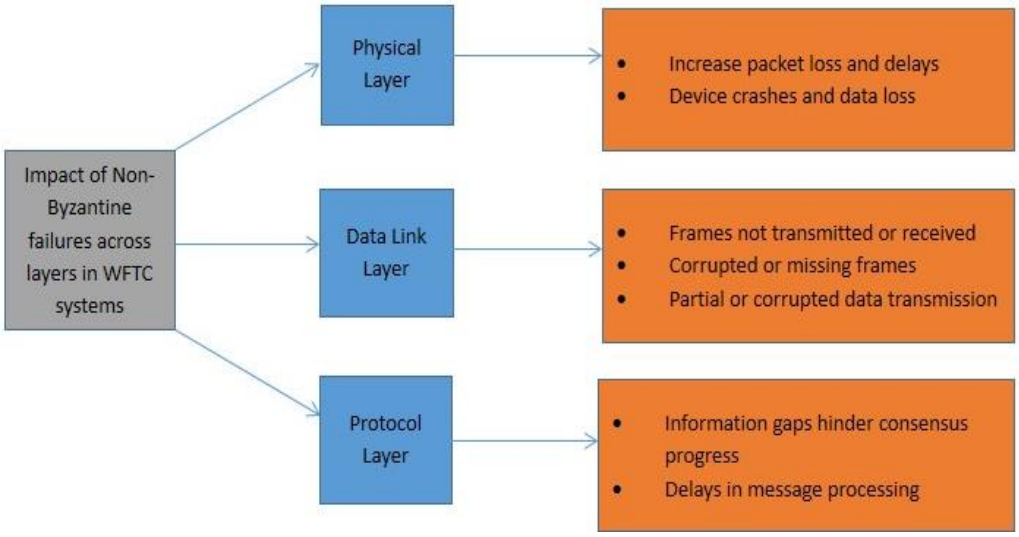

Fig3. Impact of Non-Byzantine Failures across Layers in WFTC Systems.

2) The byzantine failures in WFTC

Wireless sensor networks (WSNs) play a crucial role in various applications, from environmental monitoring to industrial automation. However, due to their distributed nature and limited resources, they are particularly vulnerable to Byzantine failures, posing a significant challenge to their reliability and trustworthiness [31].Unlike classic crash failures where nodes simply drop offline, Byzantine failures involve malicious or unpredictable behavior by individual nodes or groups. These malicious actors can wreak havoc, manipulating data, disrupting communication, and influencing critical decisions. Understanding these threats is crucial for building robust and secure WSNs [32].

This section delves into the three main categories of Byzantine failures in WFTC.

a) Malicious Attacks

Malicious attacks pose a significant threat to fault tolerance consensus in wireless sensor networks (WSNs). Unlike simple crash failures, these attacks involve deliberate attempts to disrupt the consensus process, manipulate data, or gain unauthorized control. Sybil attacks involve a malicious actor creating multiple false identities or personas within the network. These false identities can be used to disrupt routing mechanisms, manipulate sensor data, or influence the consensus process itself. By generating fake identities, the attacker aims to subvert the integrity of the system, leading to falsified information flow, distorted sensor readings, and compromised decision-making processes [33]. In a DoS attack, malicious actors flood the network with excessive traffic or requests, overwhelming the resources of sensor nodes in WSNs. This flooding of requests prevents legitimate nodes from participating effectively in the consensus protocols, disrupting the agreement process. As a result, the network experiences operational disruptions, such as delays, data loss, or inaccurate decisions [34].

b) Collusion

Groups of compromised nodes can collude to launch coordinated attacks on the consensus process. They can manipulate voting protocols, spread misinformation, or exploit vulnerabilities in consensus algorithms to gain undue influence or block decisions. By working together, these compromised nodes aim to disrupt the normal operation of the network, potentially influencing decisions in their favor or blocking legitimate decisions. Colluding nodes may exploit vulnerabilities in the consensus algorithms or underlying network infrastructure to further their malicious goals. This could involve exploiting weaknesses in encryption protocols, compromising authentication mechanisms, or exploiting implementation flaws in the consensus algorithms themselves[35]. By identifying and exploiting these vulnerabilities, colluding nodes can gain undue influence over the consensus process and potentially disrupt the normal operation of the network. Detecting and preventing collusion is challenging due to the difficulty of distinguishing between genuine and malicious behaviour.

c) Software Exploits

Vulnerabilities in the operating systems, middleware, or application software running on sensor nodes can be exploited by attackers to disrupt communication channels or inject malicious code into the consensus process. Unpatched software and outdated security practices increase the risk of such exploits. Attackers may exploit vulnerabilities in the software stack of sensor nodes to disrupt communication channels within the network. This could involve launching denial-of-service (DoS) attacks to overwhelm communication protocols or exploiting protocol weaknesses to intercept, modify, or block communication between nodes. By disrupting communication channels, attackers can undermine the consensus process and compromise the reliability of data transmission within the network. Attackers may exploit software vulnerabilities to inject malicious code into the consensus process, compromising the integrity of the data and decision-making mechanisms within the network[36]. This could involve exploiting buffer overflow vulnerabilities, injection flaws, or other software weaknesses to execute arbitrary code on the compromised nodes. Once injected, malicious code can manipulate sensor readings, falsify consensus decisions, or facilitate further attacks within the network. Fig 4. represents a detailed breakdown of network vulnerabilities and associated attacks across Protocol, Physical, and Data layers. It offers insights into potential impacts, specific attacks or vulnerabilities.

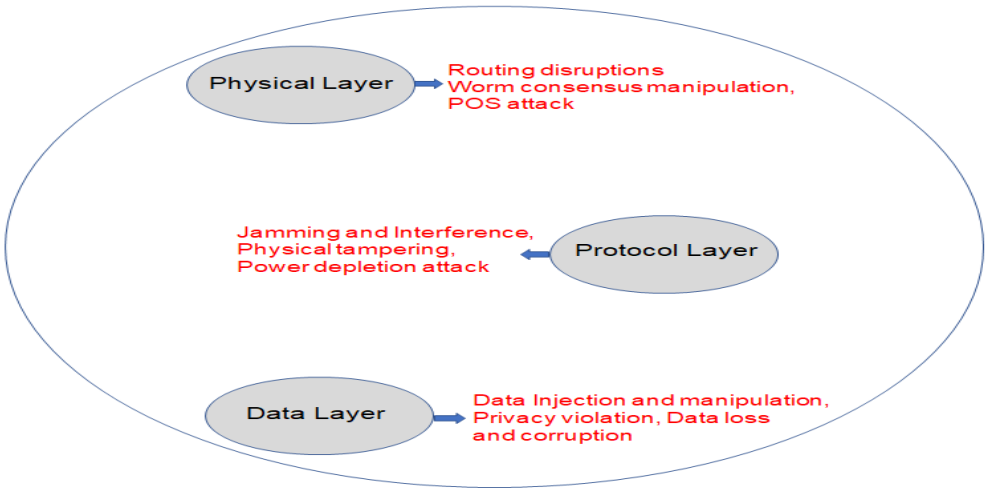

Fig 4: Analysis of Byzantine Failure Impacts and Associated Vulnerabilities across Layers.

C. NBFTC Mechanisms for Wireless Networks

Non-Byzantine Fault Tolerant Consensus refers to a consensus mechanism employed in distributed systems to achieve agreement among nodes without explicitly addressing Byzantine faults. In distributed computing, Byzantine faults encompass arbitrary and malicious behavior exhibited by a subset of nodes, such as sending contradictory messages or intentionally misleading other nodes [37]. Non-Byzantine Fault Tolerant Consensus mechanisms focus on achieving consensus in the presence of benign faults, such as network delays, node failures, or packet loss, rather than malicious behavior. These mechanisms aim to ensure system reliability and consistency despite these non-malicious failures [38]. Wireless networks, permeating diverse realms from environmental monitoring to critical infrastructure, offer immense potential but face unique challenges. Unlike their wired counterparts, they operate in dynamic environments susceptible to unforeseen disruptions like signal interference, fluctuating bandwidth, and node mobility. In such unpredictable landscapes, ensuring consistency and reliability becomes crucial for seamless communication and data integrity. This paper delves into the fascinating world of Non-Byzantine Fault-Tolerant Consensus (NBFTC) algorithms, specifically tailored for navigating the challenges of wireless networks. These algorithms empower diverse nodes within the network to agree on a shared state even in the face of unintentional errors like crashes, network delays, or message losses, safeguarding against data inconsistencies and disruptions. The journey of NBFTC algorithms began with pioneering efforts like Two-Phase Commit [39], renowned for its coordinated decision-making. However, its centralized nature and two-phase process pose limitations in highly dynamic wireless contexts, where delays and timeouts can disrupt communication. It operates in two phases: the prepare phase, where a coordinator node collects acknowledgments from participant nodes, and the commit phase, where the coordinator instructs all nodes to either commit or abort the transaction based on the outcome of the prepare phase. Raft[40] designed for managing a replicated log. It was developed as a simpler alternative to the Paxos algorithm. In Raft, nodes organize themselves into a leader-follower model, where one node serves as the leader and others act as followers. The leader is responsible for managing the replication process, while followers replicate the leader's actions. It guarantees strong consistency and fault tolerance in wireless distributed systems. Zab[41] is used in Apache ZooKeeper, a distributed coordination service. It is specifically designed for managing distributed state machines. Zab ensures that all changes to the distributed state machine are ordered and consistent across all nodes in the system. Similar to Raft, Zab also utilizes a leader-follower model, where one node acts as the leader and others as followers. The leader receives client requests, assigns sequence numbers to them, and broadcasts them to followers. Followers acknowledge these messages and replicate them in the same order. Zab guarantees linearizability and fault tolerance in distributed systems, making it suitable for coordination tasks in large-scale distributed applications. FLETA is a blockchain platform featuring the Proof of Formulation (PoF) consensus algorithm, designed for scalability and interoperability. Participants serve as Formulators or Observers, responsible for block proposal and validation, respectively. With a focus on non-Byzantine fault tolerance, FLETA offers a developer-friendly environment for decentralized applications (dApps) across various industries [42]. QUORUM was introduced to address the need for fault-tolerant consensus in distributed systems. By employing a quorum-based approach, it ensures agreement among nodes, even in the presence of non-malicious faults. This ensures system reliability and integrity, crucial for applications requiring consensus in dynamic and unreliable environments such as wireless networks [43]. These algorithms empower diverse nodes within a network to reach agreement even in the face of unintentional errors like crashes, network delays, or message losses, safeguarding against data inconsistencies and disruptions. Choosing the right NBFTC algorithm is crucial for optimal performance and security in your wireless network. Table 1 presents a comparison of several prominent NBFTC algorithms:

Table 1: Key Features and Considerations of Prominent NBFTC Algorithms.

Ref.	Algorithm	Main idea	Framework	Advantage	Limitations
39	Two-Phase Commit (2PC)	Ensures atomic commitment across multiple participants in a distributed transaction.	Centralized coordination.	All participants agree on the outcome before completing the transaction.	Cannot handle crashes during the commit phase, leading to inconsistencies.
40	Raft	Leader-based data consistency.	Leader failover mechanisms	Lightweight, efficient, resource-friendly.	Frequent communication for log replication.
41	Zab	Replicated state machine for scalability and fault tolerance.	Leader failover mechanisms.	Handles node mobility, scalable.	Resource-intensive.
42	Proof-of-Formulation (PoF)	Achieves consensus through verified formulated solutions.	Decentralized network validates proposals with cryptographic or mathematical proofs.	Enhances decentralization, security, and flexibility in achieving network consensus.	Susceptibility to Sybil attacks in consensus process.

D. BFTC Mechanisms for Wireless Networks

In wireless networks, consensus building is crucial for dependable and secure communication between nodes, particularly when there are bad or malicious actors present. Byzantine fault-tolerant (BFT) systems present a viable way

to deal with these issues. These systems ensure that nodes in a network can reach an agreement despite the presence of faulty or malicious nodes, thereby guaranteeing the integrity and reliability of the communication process.
One such innovative approach to Byzantine fault tolerance in wireless networks was the RBFT protocol, as outlined in [44], proposes a technique for concurrently running multiple instances of a Byzantine Fault Tolerant (BFT) protocol. Each instance is led by a primary replica, necessitating the support of several primary replicas for proper protocol operation. The protocol requires N nodes to tolerate up to 3f + 1 faults, where f represents the number of Byzantine error nodes capable of ensuring resilience. In RBFT, there is a distinction between primary and backup instances, with nodes only executing requests authorized by the primary instance. HoneybadgerBFT emerges as a pioneering solution, offering Byzantine fault-tolerant consensus. Employing sophisticated cryptographic techniques, HoneybadgerBFT enables nodes to reach agreement securely, even in the face of adversarial behavior. Its resilience to Byzantine faults ensures the integrity and consistency of distributed systems, making it a cornerstone for dependable communication in decentralized environments[45]. In wireless networks, HotStuff's emergence addresses the critical need for efficient and reliable consensus mechanisms. Its innovative approach to Byzantine fault-tolerant (BFT) consensus makes it particularly relevant for wireless environments where communication can be prone to disruptions and delays. By offering high throughput and low latency, HotStuff enhances the reliability of communication among wireless nodes, ensuring the integrity and consistency of distributed systems operating in wireless settings[46]. To address the demand for a simpler and more comprehensible consensus algorithm in contrast to existing options like Paxos[6], Raft[47] was introduced. It was designed to be more approachable for developers and easier to implement correctly. Raft's simplicity and clarity make it a popular choice for consensus in various distributed systems, including wireless networks, where ease of deployment and maintenance are essential factors. Tendermint, proposed as a Byzantine fault-tolerant (BFT) consensus algorithm, offers significant potential for wireless networks. Its design, primarily intended for blockchain networks, brings benefits such as secure and consistent transaction validation to wireless environments. With the rising popularity of decentralized applications and the increasing integration of wireless technologies, Tendermint's introduction addresses the need for robust consensus mechanisms suitable for wireless networks[48]. Table 2 provides an exhaustive analysis of Byzantine fault tolerance mechanisms in distributed systems, including wireless networks. This table offers invaluable insights for researchers and practitioners, enabling a thorough understanding of the advantages and limitations inherent in each approach.

Table 2: Comparative Analysis of Byzantine Fault Tolerance Mechanisms in Wireless Systems.

Ref.	Algorithm	Main idea	Framework	Advantage	Limitations
44	RBFT	Merges HBFT, RAFT for layered Byzantine fault tolerance.	The algorithm utilizes message passing and voting protocols to reach consensus.	Hierarchical structure allows for better scalability compared to traditional BFT algorithms.	The message passing and voting procedures can introduce additional overhead compared to simpler algorithms.
S45	HoneybadgerBFT	Robust cryptographic scheme for consensus.	Asynchronous.	Fast, secure, resilient to Byzantine faults.	High communication overhead and complexity.
46	HotStuff	Leader-based consensus with fast block propagation.	Partial synchronization.	Efficient, high throughput, low latency.	Require more computational resources compared to other BFT algorithms.
48	Tendermint	Separation of consensus engine and application layer.	Partial synchronization.	Secure transaction validation, fast finality.	limited deployments in wireless networks.

V. CONCLUSION

This survey paper delves into the challenges faced by wireless networks and explores Non-Byzantine and Byzantine Fault-Tolerant Consensus algorithms to ensure reliability and consistency in communication. This paper thoroughly discusses the impact of both Byzantine and non-Byzantine failures on the physical, protocol, and data layers within wireless networks. The study emphasizes the importance of fault-tolerant consensus in enhancing system resilience and performance in wireless environments Based on a structured review of fault-tolerant consensus mechanisms and comparisons of various algorithms, adopting an approach tailored to the specific situation is advisable. This allows for optimal adaptation to dynamic conditions and ensures effective mitigation of Byzantine and non-Byzantine failures in wireless network environments.

VI. FUTURE DIRECTIONS

In the realm of Wireless Fault-Tolerant Consensus (WFTC) systems, there are several avenues for further exploration and development. These include: Enhancing Byzantine fault tolerance through the creation of more sophisticated algorithms and protocols that can provide increased resilience against Byzantine failures in wireless networks. This involves exploring innovative cryptographic techniques and consensus mechanisms to counter malicious behavior and ensure the integrity of consensus protocols. Integrating machine learning techniques to strengthen fault detection and mitigation strategies in WFTC systems. By leveraging machine learning algorithms for anomaly detection and adaptive

consensus management, researchers can enable proactive fault handling and dynamic adjustment of consensus protocols based on real-time network conditions. Exploring the application of fault-tolerant consensus algorithms in edge computing and IoT environments. This involves developing tailored consensus mechanisms that address the unique challenges posed by edge devices and IoT ecosystems, ensuring reliable communication and coordination in these dynamic and resource-constrained settings. Addressing emerging threats and vulnerabilities in WFTC systems, with a focus on data security and privacy. This includes implementing robust security measures to safeguard consensus-related data from unauthorized access, ensuring data confidentiality, integrity, and availability, and mitigating potential cyber-attacks targeting consensus protocols. By exploring these avenues for further research and development, stakeholders in the field of Wireless Fault-Tolerant Consensus can advance the reliability, resilience, and performance of wireless communication systems, paving the way for more robust and efficient network operations in the future.

References

[1] Kumar, S., Velliangiri, S., Karthikeyan, P., Kumari, S., Kumar, S., & Khan, M. K. (2021). A survey on the blockchain techniques for the Internet of Vehicles security. *Transactions on Emerging Telecommunications Technologies*, e4317.

[2] Li, S., Oikonomou, G., Tryfonas, T., Chen, T. M., & Da Xu, L. (2014). A distributed consensus algorithm for decision making in service-oriented internet of things. *IEEE Transactions on Industrial Informatics*, *10*(2), 1461-1468.

[3] Ahmed, N., Kanhere, S. S., & Jha, S. (2005). The holes problem in wireless sensor networks: a survey. *ACM SIGMOBILE Mobile Computing and Communications Review*, *9*(2), 4-18.

[4] Driscoll, K., Hall, B., Sivencrona, H., & Zumsteg, P. (2003, September). Byzantine fault tolerance, from theory to reality. In *International Conference on Computer Safety, Reliability, and Security* (pp. 235-248). Berlin, Heidelberg: Springer Berlin Heidelberg.

[5] Fischer, M. J., & Lynch, N. A. (1982). A lower bound for the time to assure interactive consistency. *Inf. Process. Lett.*, *14*(4), 183-186.

[6] Clement, A., Wong, E., Alvisi, L., Dahlin, M., & Marchetti, M. (2009). Making Byzantine fault tolerant systems tolerate Byzantine faults. In *Proceedings of the 6th USENIX symposium on Networked systems design and implementation*. The USENIX Association.

[7] Lamport, L., & Lynch, N. (1990). Distributed computing: Models and methods. In *Formal models and semantics* (pp. 1157-1199). Elsevier.

[8] Onireti, O., Zhang, L., & Imran, M. A. (2019, December). On the viable area of wireless practical byzantine fault tolerance (pbft) blockchain networks. In *2019 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE.

[9] Swanson, T. (2015). Consensus-as-a-service: a brief report on the emergence of permissioned, distributed ledger systems. *Report, available online*, *28*..

[10] Zou, Y., Xu, M., Yu, J., Zhao, F., & Cheng, X. (2021). A fast consensus for permissioned wireless blockchains. *IEEE Internet of Things Journal*.

[11] Guanlin Jing, Yifei Zou, Dongxiao Yu, Chuanwen Luo, and Xiuzhen Cheng. Efficient fault-tolerant consensus for collaborative services in edge computing. IEEE Transactions on Computers, 72(8):2139–2150, 2023.

[12] Z. Zhou, O. Onireti, L. Zhang and M. A. Imran, "Performance Analysis of Wireless Practical Byzantine Fault Tolerance Networks Using IEEE 802.11," *2021 IEEE Globecom Workshops (GC Wkshps)*, Madrid, Spain, 2021, pp. 1-6, doi: 10.1109/GCWkshps52748.2021.9682068.

[13] Adday GH, Subramaniam SK, Zukarnain ZA, Samian N. Fault Tolerance Structures in Wireless Sensor Networks (WSNs): Survey, Classification, and Future Directions. *Sensors*. 2022; 22(16):6041. https://doi.org/10.3390/s22166041

[14] S. Iyer, S. Thakur, M. Dixit, A. Agrawal, R. Katkam and F. Kazi, "Blockchain based Distributed Consensus for Byzantine Fault Tolerance in PMU Network," *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kanpur, India, 2019, pp. 1-7, doi: 10.1109/ICCCNT45670.2019.8944881.

[15] Lima, M. N., Dos Santos, A. L., & Pujolle, G. (2009). A survey of survivability in mobile ad hoc networks. *IEEE Communications surveys & tutorials*, *11*(1), 66-77.

[16] Al-Nakhala, N., Riley, R., & Elfouly, T. (2015). Distributed algorithms in wireless sensor networks: An approach for applying binary consensus in a real testbed. *Computer Networks*, *79*, 30-38. *tutorials*, *11*(1), 66-77.

[17] Gao, C., He, X., Dong, H., Liu, H., & Lyu, G. (2022). A survey on fault-tolerant consensus control of multi-agent systems: trends, methodologies and prospects. *International Journal of Systems Science*, *53*(13), 2800-2813.

[18] Almeida, J., Rufino, J., Alam, M., & Ferreira, J. (2019). A survey on fault tolerance techniques for wireless vehicular networks. *Electronics*, *8*(11), 1358. IEEE.

[19] Xu, Q., Zou, Y., Yu, D., Xu, M., Shen, S., & Li, F. (2020, September). Consensus in wireless blockchain system. In *International Conference on Wireless Algorithms, Systems, and Applications* (pp. 568-579). Cham: Springer International Publishing.

[20] Kumari, P., & Kaur, P. (2021). A survey of fault tolerance in cloud computing. *Journal of King Saud University-Computer and Information Sciences*, *33*(10), 1159-1176.

[21] Liu, H., Nayak, A., & Stojmenović, I. (2009). Fault-tolerant algorithms/protocols in wireless sensor networks. *Guide to wireless sensor networks*, 261-291.

[22] Bhushan, B., Khamparia, A., Sagayam, K. M., Sharma, S. K., Ahad, M. A., & Debnath, N. C. (2020). Blockchain for smart cities: A review of architectures, integration trends and future research directions. *Sustainable Cities and Society*, *61*, 102360.
[23] Neumann, P. G. (2000). Practical architectures for survivable systems and networks. *Prepared by SRI International for the US Army Research Laboratory*.
[24] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2008). Wireless sensor networks: A survey. Computer Networks. *IEEE Communications Magazine*, 250.
[25] Zou, Y., Yang, L., Jing, G., Zhang, R., Xie, Z., Li, H., & Yu, D. (2024). A survey of fault tolerant consensus in wireless networks. *High-Confidence Computing*, 100202.
[26] Gupta, G., & Younis, M. (2003, March). Fault-tolerant clustering of wireless sensor networks. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.* (Vol. 3, pp. 1579-1584). IEEE.
[27] Gil, S., Yemini, M., Chorti, A., Nedić, A., Poor, H. V., & Goldsmith, A. J. (2023). How Physicality Enables Trust: A New Era of Trust-Centered Cyberphysical Systems. *arXiv preprint arXiv:2311.07492*.
[28] Barborak, M., Dahbura, A., & Malek, M. (1993). The consensus problem in fault-tolerant computing. *ACM Computing Surveys (CSur)*, *25*(2), 171-220.
[29] Mok, A. K. L. (1983). *Fundamental design problems of distributed systems for the hard-real-time environment* (Doctoral dissertation, Massachusetts Institute of Technology).
[30] G. Jing, Y. Zou, D. Yu, C. Luo, and X. Cheng. Efficient fault-tolerant consensus for collaborative services in edge computing. IEEE Trans.Computers, 72(8):2139–2150, 2023.
[31] Chen, R., Park, J. M. J., & Bian, K. (2012). Robustness against Byzantine failures in distributed spectrum sensing. *Computer Communications*, *35*(17), 2115-2124.
[32] Biswas, K., & Ali, M. L. (2007). Security threats in mobile ad hoc network.
[33] Tangpong, A. (2010). Managing sybil identities in distributed networks.
[34] Ramasamy, L. K., KP, F. K., Imoize, A. L., Ogbebor, J. O., Kadry, S., & Rho, S. (2021). Blockchain-based wireless sensor networks for malicious node detection: A survey. *IEEE Access*, *9*, 128765-128785.
[35] Liu, C., Zhang, X., Chai, K. K., Loo, J., & Chen, Y. (2021). A survey on blockchain-enabled smart grids: Advances, applications and challenges. *IET Smart Cities*, *3*(2), 56-78.
[36] Bhushan, B., Sahoo, C., Sinha, P., & Khamparia, A. (2021). Unification of Blockchain and Internet of Things (BIoT): requirements, working model, challenges and future directions. *Wireless Networks*, *27*, 55-90.
[37] Natoli, C., Yu, J., Gramoli, V., & Esteves-Verissimo, P. (2019). Deconstructing blockchains: A comprehensive survey on consensus, membership and structure. *arXiv preprint arXiv:1908.08316*.
[38] Shammar, E. A., Zahary, A. T., & Al-Shargabi, A. A. (2021). A survey of IoT and blockchain integration: Security perspective. *IEEE Access*, *9*, 156114-156150.
[39] Guerraoui, R. (1995). Revisiting the relationship between non-blocking atomic commitment and consensus. In *Distributed Algorithms: 9th International Workshop, WDAG'95 Le Mont-Saint-Michel, France, September 13–15, 1995 Proceedings 9* (pp. 87-100). Springer Berlin Heidelberg.
[40] Ongaro, D., & Ousterhout, J. (2014). In search of an understandable consensus algorithm. In *2014 USENIX annual technical conference (USENIX ATC 14)* (pp. 305-319).
[41] Junqueira, F. P., Reed, B. C., & Serafini, M. (2011, June). Zab: High-performance broadcast for primary-backup systems. In *2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN)* (pp. 245-256). IEEE.
[42] Maksymyuk, T., Gazda, J., Volosin, M., Bugar, G., Horvath, D., Klymash, M., & Dohler, M. (2020). Blockchain-empowered framework for decentralized network management in 6G. *IEEE Communications Magazine*, *58*(9), 86-92.
[43] Cowling, J., Myers, D., Liskov, B., Rodrigues, R., & Shrira, L. (2006, November). HQ replication: A hybrid quorum protocol for Byzantine fault tolerance. In *Proceedings of the 7th symposium on Operating systems design and implementation* (pp. 177-190).
[44] Castro, M., & Liskov, B. (1999, February). Practical byzantine fault tolerance. In *OsDI* (Vol. 99, No. 1999, pp. 173-186).
[45] Miller, A., Xia, Y., Croman, K., Shi, E., & Song, D. (2016, October). The honey badger of BFT protocols. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 31-42).
[46] Yin, M., Malkhi, D., Reiter, M. K., Gueta, G. G., & Abraham, I. (2019, July). HotStuff: BFT consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing* (pp. 347-356).
[47] Huang, D., Ma, X., & Zhang, S. (2019). Performance analysis of the raft consensus algorithm for private blockchains. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(1), 172-181.
[48] Buchman, E. (2016). *Tendermint: Byzantine fault tolerance in the age of blockchains* (Doctoral dissertation, University of Guelph).