future internet

MDPI

Review

Securing Wireless Sensor Networks Using Machine Learning and Blockchain: A Review

Shereen Ismail [1,*], Diana W. Dawoud [2] and Hassan Reza [1]

[1] School of Electrical Engineering and Computer Science, University of North Dakota, Grand Forks, ND 58202, USA; hassan.reza@ndus.edu
[2] College of Engineering and Information Technology, University of Dubai, Dubai 14143, United Arab Emirates; ddawoud@ud.ac.ae
* Correspondence: shereen.ismail@ndus.edu

Abstract: As an Internet of Things (IoT) technological key enabler, Wireless Sensor Networks (WSNs) are prone to different kinds of cyberattacks. WSNs have unique characteristics, and have several limitations which complicate the design of effective attack prevention and detection techniques. This paper aims to provide a comprehensive understanding of the fundamental principles underlying cybersecurity in WSNs. In addition to current and envisioned solutions that have been studied in detail, this review primarily focuses on state-of-the-art Machine Learning (ML) and Blockchain (BC) security techniques by studying and analyzing 171 up-to-date publications highlighting security aspect in WSNs. Then, the paper discusses integrating BC and ML towards developing a lightweight security framework that consists of two lines of defence, i.e, cyberattack detection and cyberattack prevention in WSNs, emphasizing the relevant design insights and challenges. The paper concludes by presenting a proposed integrated BC and ML solution highlighting potential BC and ML algorithms underpinning a less computationally demanding solution.

Keywords: Internet of Things; wireless sensor networks; security; machine learning; blockchain; detection; prevention; cyberattacks; integration; review

Citation: Ismail, S.; Dawoud, D.W.; Reza, H. Securing Wireless Sensor Networks Using Machine Learning and Blockchain: A Review. *Future Internet* **2023**, *15*, 200. https://doi.org/10.3390/fi15060200

Academic Editors: Christoph Stach and Clémentine Gritti

Received: 27 April 2023
Revised: 15 May 2023
Accepted: 22 May 2023
Published: 30 May 2023

1. Introduction

Wireless Sensor Networks are the backbone that enables Internet of Things (IoT) at low cost and low power [1]. These networks have been considered for a wide range of applications, such as military, environmental, healthcare, and civilian, despite being vulnerable to attacks [2]. Indeed, Wireless Sensor Networks (WSNs) result in major concerns in terms of security. Concerns include the use of devices which have resource constraints in terms of energy, the adopted wireless broadcasting channels, the involvement of multi-hop relays, the dynamic network topology, variable medium-to-large network scales, heterogeneous sensor node fabrication, and most importantly, the different routing protocols employed. Securing WSNs is relevant to securing IoT [3], as the latter comprises one or more WSNs, which implies that developing prevention, detection, and mitigation security solutions for WSNs are essential for establishing secure and reliable IoT systems.

Classical WSN security techniques, such as spread spectrum, cryptography, and key management [4,5], may not efficiently detect attacks, and can demand sophisticated software and hardware changes, rendering these solutions insufficient to address WSN security concerns, as WSN devices constrain the network's power, storage, computational, and communication capabilities [6]. There has been growing interest in novel security paradigms, with cybersecurity companies investing as much as USD 119 billion to solve these problems [7]. This has led to newly evolved means aimed at strengthening WSN security against possible cyberattacks via Machine Learning (ML) and Blockchain (BC) [8].

Compared to classical techniques, ML techniques are particularly useful in WSNs and IoT applications, as computational complexity and communication overhead can be signifi-

cantly decreased, no human intervention is required, and they perform better in dynamic environments. On the other hand, BC allows highly secure data transactions within any network similar to WSNs [9]. The fact that ML and BC can potentially provide promising solutions and effective mechanisms to protect and secure WSNs against cyberattacks has motivated several recent research works focused on evaluating the performance of BC and ML to secure WSNs. The performance of ML and BC is affected by the challenging characteristics of WSNs, such as its large generated data volume, which are extremely hard to manage, especially when considering highly dense networks. To this end, this paper attempts to answer the following overarching research questions. How is ML used to detect WSN cyberattacks? How is BC used to prevent WSN cyberattacks? How can the integration of ML and BC provide an effective framework to protect and secure WSNs against cyberattacks? Finally, What are the key technical challenges related to this integration? Thus, the main contributions of this survey are: (a) classification of WSN cyberattacks and the unique characteristics that complicate the design of effective detection and prevention mechanisms against cyberattacks; (b) a literature review of the existing Intrusion Detection System (IDS) architectures in the context of WSNs; (c) a comprehensive taxonomy of ML and BC along with an evaluation of relevant existing security techniques and challenges; (d) discussion of an integrated solution incorporating both technologies towards development of a WSN that is significantly immune against attacks; and (e) an ultimate overview of our approach to providing a lightweight and integrated ML and BC framework towards enhanced protection against cyberattacks in WSN contexts.

The rest of this paper is organized as follows: Section 2 reviews existing surveys on ML and BC solutions in the context of securing WSNs; Section 3 outlines the unique WSN characteristics that present network security challenges when developing such techniques; Section 4 illustrates the security requirements for designing a secure WSN; Section 5 classifies and defines cyberattacks that target WSNs; Section 6 discusses the underlying IDS architectures considered in conjunction with different WSN architectures; Section 7 extends the discussion to include different types of IDSs used for intrusion detection; Sections 8 and 9 focus on the respective taxonomies of ML and BC techniques used to detecting cyberattacks, along with related aspects; Section 10 explores the integration of BC and ML towards developing a lightweight security framework for WSNs and presents our approach to developing such a framework for cyberattack prevention and detection in WSN contexts; finally, Section 11 concludes this review.

2. Existing Surveys on ML and BC in WSN

This paper discusses ML and BC protection mechanisms in a comprehensive manner [10–15]; however, the emphasis is on securing WSNs. In this regard, a few previous surveys have focused on presenting state-of-the-art ML and BC techniques for WSN cybersecurity. Key surveys tackling WSN security are tabulated in Table 1, which highlights the different subtopics covered, including ML, BC, attack taxonomy, and ML–BC integration, among others. The surveyed sources were collected from popular academic databases, such as IEEE Xplore, Elsevier, and Scopus, as per the most recent citation provided by Google Scholar, and are detailed in Figure 1.

Table 1 reveals that research work on ML techniques is the primary subject of existing survey papers in the literature. A number of surveys that were published between 2012 and 2017, such as [16], did not examine WSN-related ML techniques, instead jointly discussing methods adopted in both IoT and WSN. On the other hand, surveys similar to [16–19] focused primarily on WSN. The authors of [19] considered only Denial of Service (DoS) attacks over the five TCP/IP layers. The authors of [17] provided a generalized and comprehensive review of ML techniques adopted to support WSNs against their inherent limitations, including security. The paper specifically focused on ML methods used to detect outlying and misleading measurements. The authors of [14] discussed the different types of attacks targeting WSNs and associated proposed ML solutions. Protecting WSNs using several ML methods was discussed in [16]. The authors of [20] explored using ML

techniques with WSNs, including anomaly detection, with a focus on Deep learning (DL) techniques. A different research direction was analyzed in [20,21], where the authors focused on a specific type of WSN. The authors of [20] presented ML learning techniques targeting advanced WSN systems, and [21] reviewed ML techniques to secure industrial WSN systems. The authors of [22] reviewed ML algorithms and considered using software-defined networking (SDN) as a solution that can help enhance the node efficiency, creating a new foundation for using ML schemes to secure WSNs.

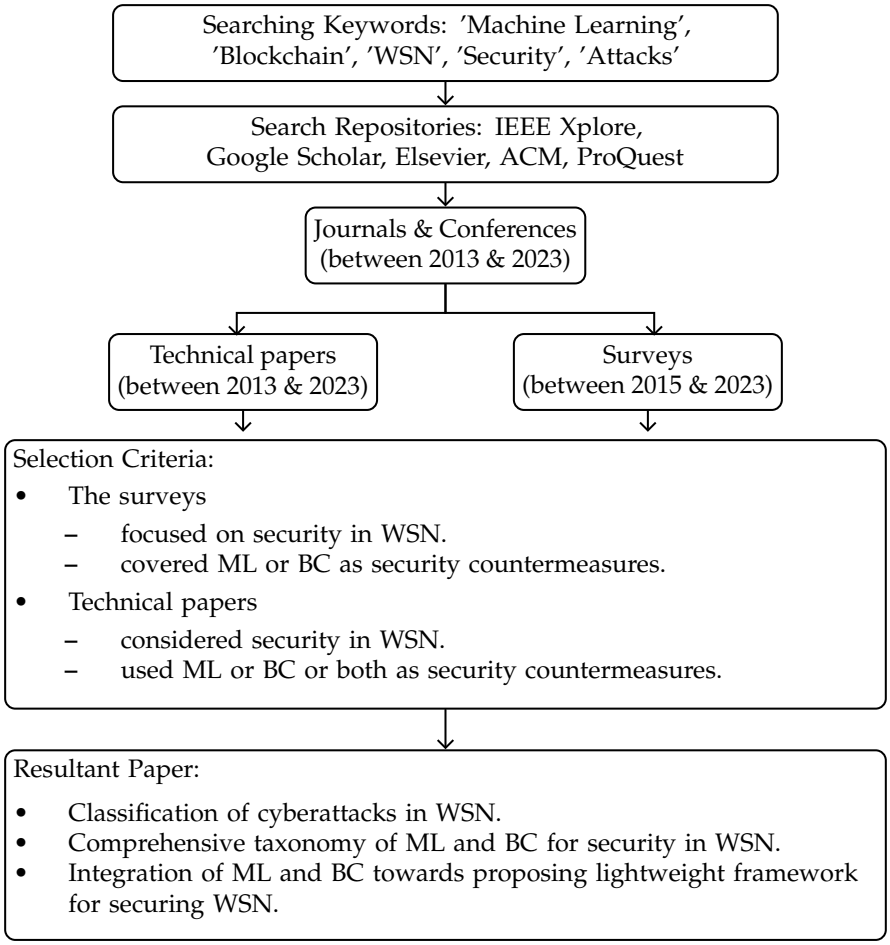

Figure 1. Paper collection criteria flowchart.

Considering BC techniques, several reviews have been conducted on securing IoT through the use of BC, such as [7,23–30]; however, Ref. [31] is the only article addressing BC for mitigating cyberattacks in WSNs. The study concluded that integrating BC techniques within WSNs has limitations, as BC is demanding in terms of both energy and computational complexity and is not expandable. Thus, to the best of our knowledge, our paper is the first work to review the integration of both technologies to improve WSN security, which is confirmed by Table 1.

Table 1. Existing works on applicability of ML and BC for WSN security.

Year	Ref.	Direction				
		ML	**BC**	**Attacks**	IDS	**Integration**
2015	[9]	✓		✓	✓	
2015	[19]	✓		DoS	✓	
2017	[32]	✓				
2018	[21]	✓		✓	✓	
2018	[16]	✓		✓		
2020	[17]	✓				
2020	[18]	✓			✓	
2021	[20]	✓			✓	
2022	[22]	✓		✓	✓	
2021	[31]		✓			
2023	our work	✓	✓	✓	✓	✓

3. WSN Security Requirements

The most important WSN security requirements include integrity, availability, scalability, non-repudiation, mutual authentication, confidentiality, and data freshness, defined in turn as follows:

1. Integrity: transmitted messages cannot be tampered with due to illegal actions when moving from one node to the other.
2. Availability: legitimate (and authenticated) nodes can effectively access the network/provided services.
3. Scalability: the network should be able to cope with increases in size and to adapt to the dynamic addition and removal of various nodes, and node functionalities must be incorporated with sensor nodes for every service without affecting the network's security level.
4. Mutual Authentication: the identities of any pair of nodes engaged in communication must be recognized before they interact.
5. Non-repudiation: the nodes cannot deny the implemented operations or alter the messages they send.
6. Confidentiality: the privacy of sensitive data transmitted over the network medium must be preserved by ensuring that any intruder or other neighboring network intercepting the communication channels cannot obtain any confidential information.
7. Data Freshness: the data must be recent in order to ensure that no old messages have been replayed and that attackers cannot confuse the network by replaying captured messages [33,34].

4. WSN Design Challenges and Unique Characteristics

WSN security solution design is highly affected by the unique features of these networks that make them more susceptible to cyberattacks than other technologies. This is primarily due to their challenging underlying infrastructure, which consists of a collection of sensor nodes utilizing scarce resources. The basic building blocks of a sensor node consist of four main units, namely, processing, sensing, communication, and power [31], as shown in Figure 2. The processing unit is the central unit, containing a processor or microcontroller that controls the sensor's activities and executes the communication protocols; however, it has limited storage memory. The processing unit is connected to the sensing unit by an Analog to Digital Converter (ADC). The sensing device captures surrounding data, which are then converted into an electrical signal by the ADC. The communication

unit typically supports data exchange between the sensor and the other network elements using a transceiver. Finally, the power unit provides the electrical energy required by the other units using limited-lifespan batteries. Optional sensor hardware additions include power generation and mobilization units [35]. Certain nodes may include a location-finding unit for positional localization in reference to the node's neighbors. These special characteristics must be identified before they can be used in the design and development of more secure networks. The following points describe the dominant design considerations in WSNs in detail, which are further highlighted in Figure 3.

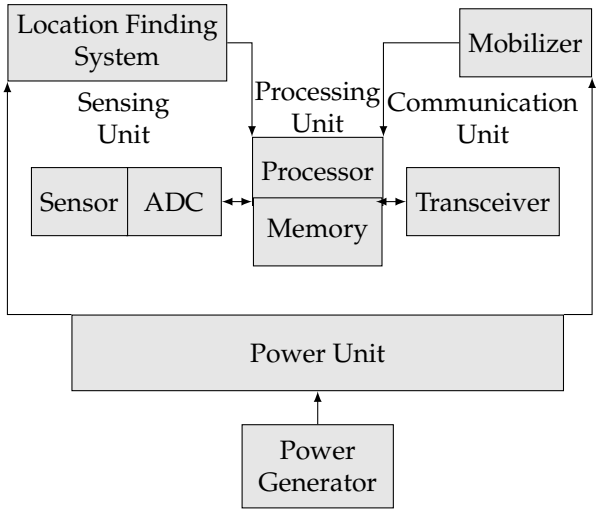

Figure 2. Illustration of sensor node building blocks.

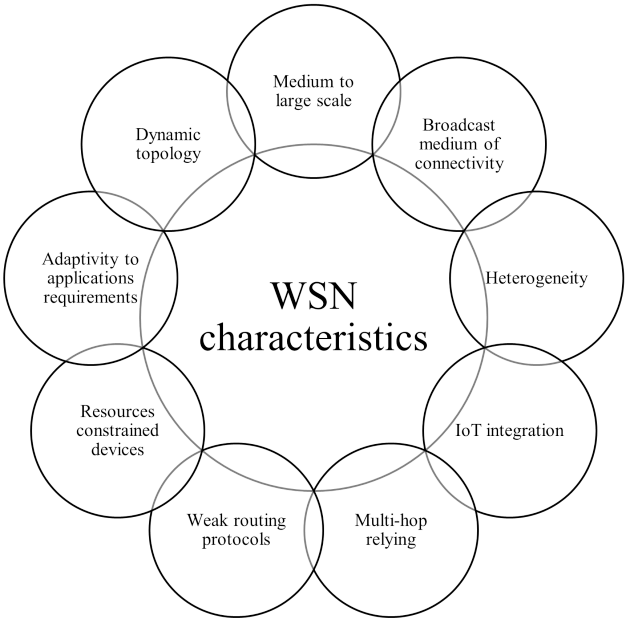

Figure 3. Unique characteristics of WSNs.

- WSNs can be used in a wide range of applications with different security requirements; however, they must be able to ensure privacy, confidentiality, integrity, freshness, and authentication.
- Sensor nodes must be heterogeneous in terms of fabrication and energy-saving strategies, such as sleep, idle, and wake-up modes, which dictates the need to provide different underlying network architectures for the different heterogeneous applications.

- WSNs have many appealing applications, creating a need for different levels of secure functionalities and service requirements, such as secure node selection, data aggregation [36], localization, and routing.
- Resource-constrained devices have limited memory, power, and transmitting bandwidth. For example, TelosB [37] is an ultra-low-power sensor with a a 16 bit processor and 8 MHz RISC microcontroller with only 10 Kb RAM, 48 Kb program memory, and 1024 Kb flash storage. The required total space for a typical code, such as TinyOS, which is the de facto standard operating system for wireless sensors, is approximately 4 Kb [38]. Therefore, any implemented security algorithm within the network must not be computationally demanding beyond these limitations.
- Security algorithms must be able to manage unsupervised sensors, which could be exposed to physical attack by demolishing the hardware or to attackers equipping sensors with extra hardware to perform hidden or malicious functions prior to their being deployed in the network area.
- Determining the adopted broadcast dynamic channel used as a wireless communication medium is challenging, as it is unattended and might be affected by collision and interference issues. WSN communication links are usually based on the 802.15.4 standard, and can be implemented via the use of other technologies as well, such as Bluetooth, ZigBee, PLC, WiFi, 4G, and 5G.
- The lack of fixed physical infrastructure is a significant design challenge due to the rapidly changing connectivity between nodes.
- A dynamic underlying network topology results from node failure, deployment of new nodes, possible variations in node position (which is especially the case under harsh environmental conditions), node mobility. The resulting flexibility in terms of link connectivity presents a design challenge for security algorithms, which must be able to adapt to network node variations in order to obtain the extra measure of protection provided by monitoring of corrupted nodes.
- WSN routing protocols have weaknesses, including malicious routing information injection, alteration, or spoofing, which might lead to network disruptions such as creation of routing loops, broadcasting of fake error messages to partition the network, attracting or repelling network traffic from particular nodes, extending or shortening route paths, and increasing end-to-end latency. These issues are likely to complicate the design of security routing techniques [39].
- Medium- to large-scale networks of hundreds or thousands of nodes deployed randomly or uniformly throughout the network field presents a challenge when designing security algorithms that are sufficiently flexible to support different security-level requirements.
- The scalability of WSNs implies handling large amounts of data that may have inconsistent, noisy, erroneous, redundant, and missing values, which requires designing intelligent security approaches that can correctly interpret data to drive intelligent decision-making.
- Data transmission over multi-hop relaying creates a significant threat, as relays could be eavesdroppers [40], and communicated data may be breached, tampered with, or forged.
- Time synchronization is an issue, as nodes are independently controlled in the field. Local clocks should be coordinated to avoid synchronization uncertainties, which could cause sensed data to become ambiguous and unreliable.
- Unexpected and unusual sensor behavior patterns may arise during WSN deployment in unpredictable and hazardous environments, potentially changing the entire historical pattern of the sensed data.

These characteristics render a completely secure WSN system almost impossible to establish, unlike its counterpart networks. The characteristics of WSN systems limit the available security options, including those similar to heavyweight classical security approaches such as spread spectrum, cryptography, and key management at either the

device level or the overall network level. These options are demanding in terms of the resources required to protect the network. As existing security solutions for WSNs are insufficient due to these networks' unique characteristics, it is difficult to create lightweight and effective security mechanisms that can enable optimization of node resource usage while supporting network scalability and without compromising security, allow for a dynamic network topology with different possible configurations and node localization, and integrate heterogeneous hardware and software platforms for sensors to allow them to detect malfunctioning or faulty nodes.

5. Cyberattacks in WSN Contexts

Cyberattacks are the greatest challenge facing communication networks worldwide. The threat of cyberattacks affects any network's connectivity, availability, reliability, and confidentiality, limiting its efficient use. Mitigating this challenge is essential, especially because the frequency and the nature of attacks have increased tremendously over time [41]. For this reason, cyberattacks targeting WSNs have been the focus of several recent studies in the literature [4,42–46]. Cyberattacks occur when good nodes are communicating over a communication link and intruder or eavesdropper nodes interfere with or disturb that link. This malicious activity usually aims to obtain, alter, or prevent the flow of data within the network using different means; therefore, this activity should be prevented, detected, and mitigated in order to maintain a reliable communication channel [47]. Malicious acts targeting WSNs have been classified in the literature in different ways: the first classification divides attacks into active or passive attacks; the second classification is based on the physical location of the attack relative to the network's physical position, using this distinction to divide attacks into inside or outside attacks; and the third classification is based on the disrupted stack Open Systems Interconnection (OSI) layer, dividing attacks into physical layer, data link layer, and network layer attacks [42,48]. Table 2 classifies a selection of classical attacks targeting WSNs and provides their definitions.

Table 2. Classification of cyberattacks on WSNs.

Attack Type	Affected Stack Layer	Attack Name	Definition
Active	Multi-layer	Man-in-the-Middle	A malicious node intercepts a message passing between two sensor nodes with the aim of modifying, injecting, or deleting content before relaying the message again.
		Denial-of-Service	An attacker performs malicious activities to prevent original users from accessing system resources.
		Distributed Denial-of-Service	A more powerful version of DoS attack that overwhelms the targeted nodes with excessive messages to exhaust their resources, leading to a system overload that prevents it from answering some or all legitimate messages.
	Application	Deluge	An attacker tries to remotely reprogram a sensor node.
		Misdirection	An attacker forwards packets to the wrong destinations or paths by misdirecting packets or altering routes towards a malicious node.

Table 2. *Cont.*

Attack Type	Affected Stack Layer	Attack Name	Definition
		Clock skewing	Disrupts sensors that requir synchronization for successful communication; an attacker desynchronizes sensor clocks by generating false timing information, leading to desynchronization of the victim nodes.
		Selective Forwarding	Malicious nodes drop a portion of a received message while forwarding most of the message, impacting data integrity.
	Transport	Flooding	An attacker sends a large number of useless packets to a legitimate node, preventing it from communicating normally and consuming its resources.
		Session Hijacking	An attacker exploits a valid session, pretends to be a victim node, and obtains fake access to the session.
		De-synchronization	An attacker intercepts sequence numbers or controls flag packets that it attempts to forge; if the attacker can desynchronize two communicating nodes, the receiver node must request retransmission from the sender for the lost packet. Frequent retransmission consumes network resources and increases traffic over the network.
	Network	Reply	An attacker records the messages sent between nodes and re-transmits them later to waste the target node's resources.
		Selective Forwarding or Grayhole	A malicious node selectively, constantly, or randomly drops packets while forwarding the remaining packets to a particular destination, which happens when relay nodes do not forward messages they receive.
		Neglect and Greed	A special case of selective forwarding attack in which the attacker arbitrarily drops some of the received packets while acknowledging the source node (neglect attack) or sends its own packets with higher priority to other nodes (greed attack) [49].

Table 2. *Cont.*

Attack Type	Affected Stack Layer	Attack Name	Definition
		Homing	An attacker analyzes traffic using a traffic pattern analysis algorithm to recognize the nodes with special responsibilities, such as cluster heads (CHs) or base station (BS), which are the attack targets. Afterwards, additional DoS attacks may be launched toward these nodes to jam or destroy them.
		Spoofing	An attacker forges its identity by impersonating another node and falsifying the identity field in routing messages to launch DoS attacks by injecting fraudulent data packets, such as falsely advertising services to other nodes or providing incorrect routing and control information to compromise network operation [50].
		Blackhole	A malicious node, usually located in the center, does not forward traffic and drops the packets completely.
		Wormhole	A collusion-based attack in which two or more malicious nodes create a low-latency data delivery tunnel between two or more malicious nodes to perform other attacks, such as a blackhole attack. For instance, the nodes may establish a low-latency tunnel by which one malicious node misroutes the packets to be forwarded and sends them to its partner using a faked routing path to disrupt routing operations in the network.
		Sybil	A single attacker node assumes several identities or steals them from other authorized nodes to create several sybil nodes that can be virtually present in different neighborhoods, then attack the network to cause problems with multipath routing, network topology, storage access, and detection [50].
		Sinkhole	A malicious node identifies itself as a blackhole to attract network traffic. The attacker observes path requests and falsely offers the shortest or most power-efficient paths to the BS. As the attacker is in the relay path between the communicating nodes, it is able to change or alter the packets passing between them [44].

Table 2. *Cont.*

Attack Type	Affected Stack Layer	Attack Name	Definition
		Hello Flooding	An attacker broadcasts advertisement 'Hello' messages with high power, asking network nodes to join an existing WSN and tricking the nodes into believing that it is located in their neighborhood. The nodes choose to route their packets through the attacker, which has a longer transmission range than normal nodes, leading to additional delays and energy waste.
	Data link	Collision	An attacker sends signals while another node is transmitting a message, causing interference that alters data packets or causes them to be considered invalid. Collision usually occurs when multiple nodes transmit data at the same frequency and data rate.
		Denial of Sleep (Sleep Deprivation)	A Malicious node prevents legitimate nodes from entering low-power sleep mode, causing them to keep wasting their energy [51].
		Power Exhaustion	In order to drain the victim node's power, an attacker sends packets over the channel continually by requesting calculations or the receipt or transmission of unnecessary data, which leads to starvation. The source of the attack can be a PC or laptop.
		Unfairness	A malicious node continuously sends packets without waiting a reasonable time to let other nodes use the channel. This is a kind of exhaustion-based attack which disrupts equal load sharing in the WSN.
	Physical	Jamming	An attacker sends a radio signal that interferes with the sensor network's use of certain radio frequencies.
		Physical or Node Tampering	An attacker physically accesses a compromised node and takes over the control, for example, to obtain sensitive information such as transmission keys [52,53].
		Node Replication or Clone	An attacker captures a compromised node, obtains access to the stored credentials, purposefully duplicates the node's identity, and then deploys clones in key positions of the WSN [54] to initiate different internal attacks.

Table 2. *Cont.*

Attack Type	Affected Stack Layer	Attack Name	Definition
Passive		Camouflage Adversaries	A camouflaged node deceives the other nodes and attract packets from them in order to either misroute the packets or eventually drop the packets.
		Eavesdropping and Traffic Analysis	The most common attack on privacy, also called sniffing or snooping, where an attacker simply discovers the content of communications.
		Passive Information Gathering	If the content of messages from network communication media, such as message identification numbers (IDs), nodes locations, and timestamps, is not encrypted then an attacker with the appropriate receiver can collect and observe the information.
		Replay or Duplication	An attacker copies a stream of messages between communicating nodes, then replays the stream to one or more of the nodes [55]

Active attacks threaten network integrity and reduce availability by continuously attempting to modify the content of the network packets or flooding the victim nodes with surplus packets. The different types of active attacks are based on the underlying stack layer disrupted by the attack, as shown in Figure 4 [42,48]. Attacks such as link jamming, physical tampering, or node replication are hardware-oriented attacks that affect the node's physical layer. These attacks are more likely to occur when the sensor is exposed to a harsh environment or open to an adversary; therefore, they are unlikely to occur when the sensor node is placed in a secure indoor location. Other attacks, such as collision, exhaustion, and unfairness, are executed against the Media Access Control (MAC) protocol at the data link layer. These attacks cause collisions that result in packet re-transmission; therefore, copies of the same packets must traverse the network, overwhelming the communication channel and wasting limited sensors energy. The most common attacks, such as sinkhole, wormhole, blackhole, selective forwarding (grayhole), 'Hello' flooding, sybil, spoofing, and altered or replayed routing information attacks, all interrupt the network layer. These attacks prevent proper packet delivery to the destination through methods such as taking advantage of the multi-hop routing protocol, in which any node routes passing through malicious nodes are unable to deliver packets or are intentionally redirected to incorrect nodes. Examples of attacks impacting the functionality of the transport layer include session hijacking, flooding, and de-synchronization attacks. For example, flooding results in node failure, as the attacker consumes node resources by sending multiple connection requests. Attacks that target the application layer include selective forwarding, deluge, and clock skewing. The most difficult to detect among these attacks is selective forwarding, as the attacker does not block packet forwarding entirely, and only drops or alters some of the received packets from selected nodes. Deluge allows the sensor nodes to be reprogrammed remotely, and clock skewing disrupts those sensors that require synchronization for successful communication. Unlike active attacks, passive attacks do not affect network integrity, instead compromising network confidentiality. These attacks sniff and read unauthorized messages through the communication channels between nodes without disrupting their communication or interrupting network processes. Passive attacks may make the network

more vulnerable to other kinds of attacks, such as camouflaged adversaries, physical tampering, eavesdropping, and traffic analysis.

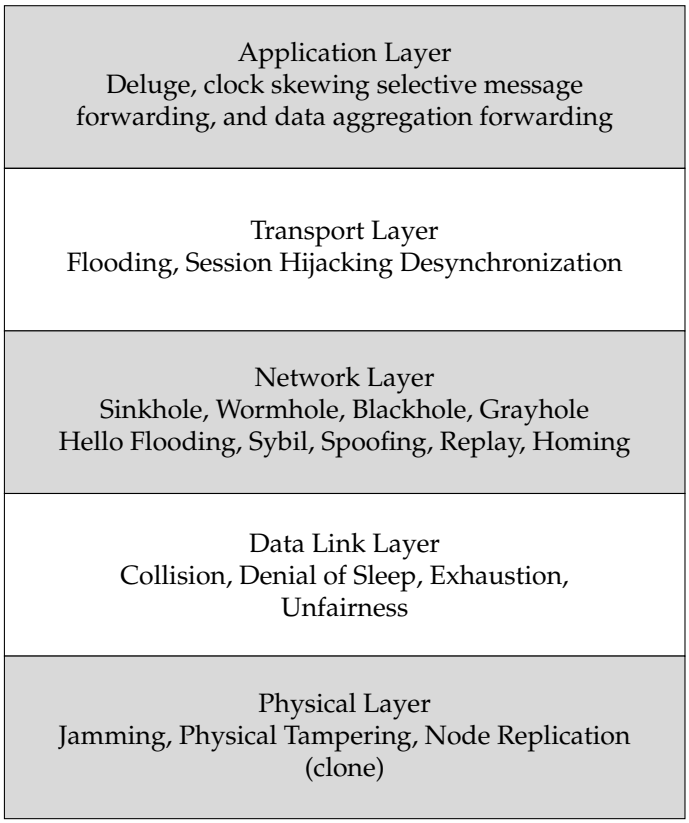

Figure 4. Active attack classification according to OSI stack layer.

Internal attacks are initiated from within the network's physical boundaries. These attacks control and utilize other network nodes to execute malicious acts. An inside attack can obtain the network transmission key or other network information from the transmitted packets within the network, then use this information to attack the entire network. A typical example of an internal attack is when an attacker takes advantage of a dump security implementation at an unsecured sensor node or a non-updated device's firmware, which allows the attacker to turn sensing devices into malicious nodes. The attacker then utilizes the node's network connectivity with other nodes to extract network data using eavesdropping, interfering, or misrouting. External attacks are initiated from beyond the network boundaries; therefore, they cannot obtain network information, such as node identification numbers or transmission keys, making attack recognition easier [48]. In addition, external attackers require powerful wireless transceivers to listen to data packets inside the network in order to accomplish malicious activities such as eavesdropping, replay, injection, and interference. Figure 5 depicts scenarios for external and internal attacks targeting a WSN.

In terms of OSI layers, the physical and network layers experience the most threatening attacks. The physical layer possesses a broadcasting channel and a dynamic topology, which allows attackers to easily listen to or sniff the communication channel and establish attacks. While, the network layer has a weak routing protocol that attackers can exploit to execute malicious acts. Another form of attack can be initiated over several WSN stack layers; such attacks across multiple layers include DoS and Man-in-the-middle (MITM) attacks [38]. DoS attacks are numerous, and include jamming and node tampering at the physical layer, collision, exhaustion, denial of sleep, unfairness at the data link layer, homing, blackhole, grayhole, wormhole, sinkhole, spoofing, 'Hello' flooding, TDMA scheduling, sybil, and replay attacks at the network layer, as well as flooding and desynchronization at the

transport layer [56]. MITM attacks work as a relay between two victims [6]; this type of attack can be passive, where the attacker eavesdrops or intercepts the data traveling on the network between two legitimate nodes without altering the data, such as eavesdropping at the physical layer, or it can be active, where the attacker can delay, drop, or modify the content of a packet, such as a replay attack at the network layer [57].

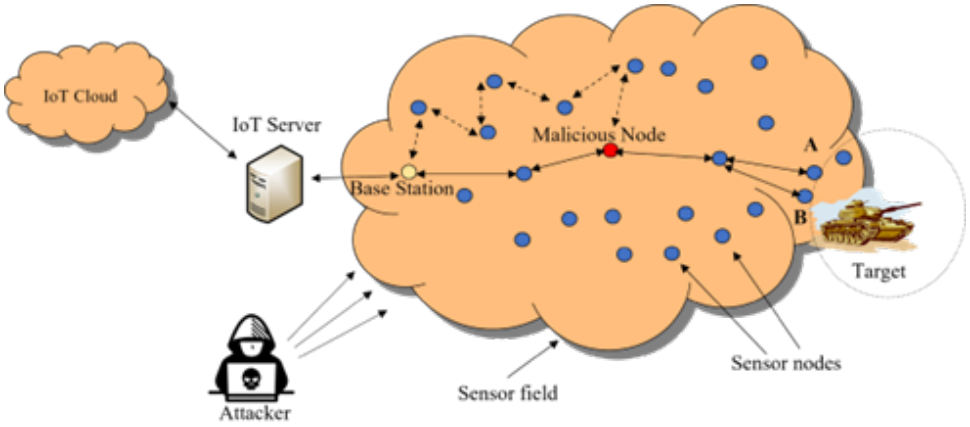

Figure 5. Internal and external cyberattack scenarios in WSN contexts.

6. Architecture of WSN vs. Architecture of IDS

Intrusions are similar to attacks in that they aim to disturb the network's normal operation or obtain access to the network's information. The IDS is the network's line of defense, designed to detect violations and tell the controller, or BS, to react appropriately.

6.1. Naive or Flat-Based WSN Architecture for Centralized IDS

In a centralized architecture, better known as a Naive WSN architecture, a central BS collects all the information sensed by all network nodes and forwards the collected information to the cloud IoT server. Similarly, in a centralized IDSs, the BS acts as a global reference that performs computationally demanding tasks to monitor and filter data traffic to facilitate attack detection. Several studies have considered executing the IDS at both the BS level and at the remote server level connected to the IoT cloud, which is called a multi-layer IDS scenario. This approach has multiple limitations, including attack detection latency, considerable communication overhead, and high energy consumption. Latency occurs when data traffic analysis is delayed until the information reaches the BS. Communication overhead is caused by the need to transmit all sensed information to the central BS over the communication link, increasing energy consumption as the node's distance relative to the central unit increases [58]. Due to these limitations, centralized IDS architectures are typically used only in very small networks.

6.2. Naive or Flat-Based WSN Architecture for Stand-Alone IDS

The opposite philosophy to centralized IDS is stand-alone IDS, which is a node-centric architecture. Each node individually uses an IDS detection model to detect any possible attack locally without needing to exchange any information with the adjacent nodes or a central BS unit. This approach does not exhibit latency when detecting node attacks or introducing communication overhead; however, energy consumption at the node level is higher than in a centralized IDS, and the nodes have lower battery life.

6.3. Naive or Flat-Based WSN Architecture for Distributed or Cooperative IDS

This approach assumes that each node has its own local IDS model to monitor the data traffic, then involves all network nodes in deciding whether an intrusion is present in the network based on the detected indicators. If a locally measured indicator is weak or inconclusive, the involved node can initiate a cooperative global intrusion detection proce-

dure in which all network nodes cooperatively participate in reaching a global decision. Otherwise, if an intrusion is locally detected with sufficient evidence, the involved node can independently alert the rest of the nodes to the presence of a violation in the network. This approach reduces false attack stimulus events, which relate to scenarios in which a violation alarm is triggered even though no real threat is in progress within the network. In this approach, node power consumption is higher and node battery life is lower than the stand-alone IDS due to the an additional optional cooperative procedure.

6.4. Naive or Flat WSN Architecture for Agent-Based IDS

Agent-based IDS involves installing the detection model in a selected subset of sensor nodes, which are called Monitor Nodes (MNs), to reduce the detection overhead faced by the stand-alone and distributed approaches. In tis approach, selected nodes perform detection in addition to their normal sensing, communication, and routing activities in the case of flat WSN architecture. Agents' tasks are relocated to another predefined subset of nodes after a certain period of time or when performing a specific mission, which improves IDS detection efficiency and increases network lifetime. Agent-based IDS is especially suitable for WSNs, as nodes near the BS can be excluded from communicating all of their samples when developing the reference ML model at the BS because they do not contribute much to the determination of hypersphere of the developed ML model. Agent-based IDS is typically preferred over centralized IDS architecture, especially for networks with geographically dispersed nodes, as in a centralized approach the nodes consume more power when transmitting their data to the central location.

6.5. Hierarchical WSN Architecture for Distributed or Cooperative IDS

A WSN's hierarchical architecture is a variation of centralized architecture, which can be implemented as cluster-based or tree-based. In a cluster-based architecture all sensor nodes are partitioned into clusters, whereas in a tree-based architecture the nodes are partitioned into trees according to their topographical area. The nodes in a tree-based architecture are organized into a routing tree rooted at the BS. Cluster-based architectures can be static or dynamic. In static clustering the sensors are divided proactively into several clusters at the time of network deployment, while in dynamic clustering the formation of clusters is triggered reactively by detecting the event of interest. In a distributed IDS, the detection model is placed in every sensor network node, allowing nodes to collaborate in order to detect possible intrusions. The clear advantage of implementing a combined hierarchical and distributed architecture is that the communication overhead is significantly lower than in other approaches, as both hierarchical and distributed architectures involve less communication exchange between nodes [58]. A disadvantage of this approach is the need for each network node to have sufficient energy, processing, and storage capacity. Studies have considered using multi-layer instead of distributed IDS, with heterogeneous detection models placed only at the BS and CHs.

7. Types of IDS

IDS-based mechanisms are effective and lightweight solutions for detecting abnormal behavior in WSN sensor nodes. An IDS requires an IDS agent or detector node that analyses the network traffic to detect a abnormal behavior. Intrusion detection at the IDS agent level involves three phases: collection, processing, and action. Network data traffic is collected during a specific time period, then this collected information is processed according to a particular detection mechanism. Detection approaches can be classified as misuse-based, anomaly-based, and specification-based detection. In misuse-based or signature-based detection, the system searches for specific patterns or signatures to identify and detect an intrusion. This approach easily detects known attacks, but cannot detect new or unknown attacks. In specification-based detection, a set of rules or specifications have been set as a reference for normal system operation; any deviation from these specifications triggers an abnormal behavior alert, allowing the system to take proper preventive actions accordingly.

This approach has a low false positive rate; however, developing the required specifications is very time-consuming. Anomaly-based detection systems learn the normal behavior profile from normal network traffic and create a reference model accordingly. This model is then used to detect any deviation from the learned pattern or behavior exceeding a certain estimated threshold for use in identifying intrusions [32].

Anomaly-based detection is adaptive, and can detect new and unknown attacks efficiently; however, it has a higher false positive rate compared to previous approaches, as any deviation from the normal behavior profile is considered an attack even though it might be due to normal activity of an unlearned profile or a faulty node producing abnormal activities [15]. Especially in critical infrastructure applications, these types of anomalies are just as harmful as those caused by intruders, and should be identified by the developed reference model [32]. Anomaly-based detection is practical, flexible, computationally feasible, bandwidth (BW) and both spectrally and memory efficient [21]; therefore, it is widely used to secure WSNs. For this reason, the focus of this survey is on anomaly-based detection.

Anomaly-based detection techniques are classified into statistical and ML approaches. The stochastic network behavior in normal conditions is measured during a specific time window and is used to establish a baseline for future detection of patterns that are different from normal traffic [58]. However, the approach continuously generates other reference profiles with a given score for comparison to the reference profile during traffic monitoring. In this approach, the IDS is able to detect an anomalous occurrence if the score is above a certain threshold. On the other hand, ML approaches use classification algorithms to detect intrusions and malicious activities. ML classification algorithms build models capable of classifying packets to distinguish between normal and abnormal packets through training. The model is installed at the sensor level, and can classify upcoming packets after training. The advantage of ML is the ability of models to learn from experience without being explicitly reprogrammed, allowing them to be improved automatically [15,59].

8. ML and Cyberattack Detection

ML algorithms are used to build self-learning classifiers consisting of behaviors, which are able to act without human intervention by using mathematical techniques based on specialized datasets. These algorithms enhance the network nodes' ability to learn without being explicitly programmed. Such models are used to make future predictions based on new input data. ML algorithms are currently used in various applications, such as smart cities, energy, agriculture, intelligent transportation systems, industry and manufacturing, search engines, social media, cyberattack detection, spam email filtering, and recommendation systems. Different ML techniques are used to improve the functionalities of WSNs, such as data sensing, CH selection, routing and optimal path determination, data aggregation, minimizing packet delivery latency, duty cycle management, quality of service (QoS) provisioning, resource management, and to increase network lifetime. ML algorithms have been used to design lightweight detection and mitigation systems to secure WSNs against cyberattacks. They allow sensor nodes to detect possible attacks and immediately take appropriate actions to mitigate the impact of an attack by triggering an alarm, determining the degree of the risk, and isolating the attacker node from the next round of network progress [60]. The ML pipeline spans data collection and pre-processing feature selection, model training using proper ML algorithms, hyperparameter tuning, model testing, validation, and deployment.

8.1. ML Methodology

Several studies have developed and investigated effective ML techniques for cyberattack detection and mitigation. Figure 6 illustrates the generalized methodology of an ML algorithm applied to ML-based IDS. The workflow includes several phases corresponding to dataset collection, data preprocessing, features selection and extraction, ML model training, hyperparameter tuning, and model testing and validation. The first step is the

availability of a dataset, which can be balanced (using of an equal number of samples for each attack type in addition to normal class samples) or imbalanced (consisting of an unequal distribution of the classes in the dataset). The next step is data preprocessing, which involves several stages: class rebalancing and sample size reduction, missing value imputations, cleaning or feature removal, data normalization, and transformation (i.e., encoding labeled data). The advantage of balancing the dataset before using it in training is to avoid bias towards the majority class. This is followed by feature selection, which involves determining an optimal set of features to help reduce dataset dimensionality, especially when considering a large dataset that may have irrelevant, redundant, erroneous, and correlated features. A lower number of dataset dimensions lead to less computational and training time being required. The reduced dataset is then utilized to train the ML model. Optimal hyperparameter values can be obtained by applying efficient tuning techniques. The final step is testing and validation, which entails using several evaluation metrics to assess ML model performance, such as the probability of detection P_d, probability of false alarm P_{fa}, probability of misdetection P_{md}, positive prediction value PPV, accuracy (ACC), F1-score, root mean square error RMSE, and receiver operating characteristics (ROC).

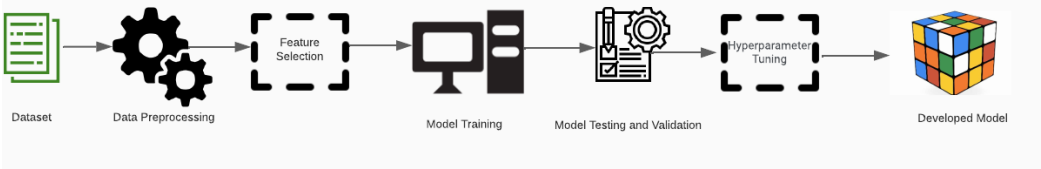

Figure 6. Illustration of generalized ML conceptual methodology.

8.2. Existing ML-Based approaches

8.2.1. Classical Machine Learning

ML algorithms are typically categorized as supervised, unsupervised, semi-supervised, and reinforcement learning. Supervised ML algorithms learn the inputs and their corresponding outputs to perform the learning process. Supervised algorithms are subdivided into regression and classification; well-known models include Logistic Regression (LR), K-Nearest Neighbors (K-NN), Support Vector Machine (SVM), Decision Trees (DT), Gaussian Naive Bayes (NB), Artificial Neural Network (ANN), and Random Forests (RF). Unsupervised learning ML algorithms only use the inputs while learning, as the associated outputs are not provided; the learning process is performed by classifying the provided input data into groups called clusters, and any new input is classified within its corresponding group. Clustering and dimensionality reduction are the two main categories of unsupervised learning. Semi-supervised learning works by combining a small amount of labeled data with a large amount of unlabeled data. In reinforcement learning, neither the inputs nor their corresponding outputs are provided, and the relationship between the input and the output is learned by interaction with the surrounding environment and a reward scheme. The reward scheme depends on the learning algorithm's performance when achieving a certain task such that a reward is provided if it achieves high performance. A popular example of reinforcement learning is Q-learning.

Several studies have discussed the efficiency of using classical ML techniques to tackle different cyberattack types [47,60–71]. For instance, research has examined well-known network layer DoS attacks (blackhole, grayhole, flooding, and TDMA scheduling). Another study used ANN and SVM to target common MAC layer attacks (collision, unfairness, and exhaustion) [72]. Other research work has considered the efficiency of RF techniques for the detection of physical layer clone attack [73]. The authors of [74] used a reinforcement learning (RL)-based IDS to detect DoS, remote-to-local, user-to-root, and probe attacks.

8.2.2. Deep Learning

DL requires a larger amount of data samples; therefore, more processing time and power are required than with classical ML techniques, which is not favorable in resource-

constrained WSN contexts. DL models are more suitable for classification and prediction tasks in IoT applications that generate unstructured data, such as images, audio, and video.

DL techniques such as recurrent neural networks (RNN), deep belief networks, and Convolutional Neural Network (CNN) are largely used for security preservation and attack detection, due to their fundamental constraints when applied to WSNs. The computational complexity associated with their training, inference, and adaptation makes their use in sensor devices impractical. Several studies have been conducted on using DL techniques, such as [72], where the authors used autoencoder neural networks with a single hidden layer of neurons for lower complexity, which suits resource-constrained WSN contexts. The authors of [75] proposed a hybrid DL model using CNN and long short-term memory (LSTM) for blackhole and grayhole attack detection. The same techniques, CNN and LSTM, were used by the authors of [76] to detect DoS attacks. The authors of [77] investigated the performance of different DL techniques, including Deep Neural Network (DNN), CNN, RNN, and CNN, in combination with RNN for a single detection layer against DoS attacks. The authors of [78] proposed a DL model using a restricted Boltzmann machine with different numbers of hidden layers. The authors of [79] proposed a DL using CNN for the detection of DoS, UR2, R2L, and probe attacks. They proposed a hybrid algorithm consisting of the whale optimization technique and artificial bee colony optimization technique. Overall, both ML and DL techniques are promising for efficient IDSs in WSNs and IoT thanks to their ability to process high–dimensional data, extract useful features from network traffic payloads, and determine complex nonlinear relationships between inputs and outputs to enable informed and intelligent decisions on the part of networks.

8.2.3. Deep Reinforcement Learning

Adapting to new or constantly evolving attacks is a major drawback in classical ML and DL algorithms due to their dependence on the fixed features of existing attacks provided by the dataset for the learning process, which limits the implementation of algorithms in applications that are vulnerable to dynamic intrusions [80]. Research activities have searched for a more efficient solution by integrating DL methods with RL, which has proved effective in various IDS applications for detecting sophisticated types of cyberattacks, especially in real-time and adversarial environments [80]. For instance, attacks that affect both the physical and MAC layers were effectively detected using a proposed deep reinforcement learning (DRL) model that relied only on partial observations. In [81], a new DRL-based IDS for WSNs was designed considering link invulnerability and node importance.

8.2.4. Federated Learning

Federated Learning (FL) supports a distributed approach to perform model training at the sensor node, unlike ML or DL. WSN nodes sense and collect the data readings, then use the locally collected data for model training [82]. Afterwards, the full locally obtained model parameters in the network are shared with a powerful node, referred to as an aggregator, usually the IoT cloud server. The aggregator then merges the received trained model parameters and generates a global model that is deployed to all WSN nodes. A system based on FL structures is more robust and privacy-preserving than a traditional ML- or DL-based IDSs, because the sensor nodes collaboratively build a global learning model while safely preserving all training data locally at the sensor storage location. In a traditional ML- or DL-based IDSs, large volumes of raw data are continuously transmitted from sensors to the BS, which involves significant channel interference and energy consumption, keeping in mind that only a small fraction of the data readings are anomalous.

Recent studies have addressed the challenges of applying FL in the context of WSNs, as FL requires additional overhead and complexity, which may affect detection accuracy and convergence speed. Anomalous samples represent a very small fraction of the local data, meaning the accuracy of the training process is reduced because only the node's locally collected data is used for training, and the local dataset may lack enough training

data for certain types of attacks. The node's resource heterogeneity and dynamic physical topology could lead to unexpected inconsistencies during the training process. Different nodes collect different numbers of data samples for training, meaning that attacks might only appear in very few nodes, and the same type of attack may have diverse distribution patterns at different nodes. This imbalanced distribution of data can slow down the training process at the aggregator and reduce performance due to diverging weights; thus, reducing the number of rounds required or the learning process to reach convergence is necessary in the context of a WSN in order to reduce power consumption.

Fast iteration convergence is a challenge when considering that training data samples at the local nodes are not independent and identically distributed (iid) in FL, as is the case with other ML techniques. This challenge is caused by issues such as non-uniform placement of sensors in space, faulty sensors, and high packet loss rates. Despite this, several studies support the assumption that the data samples collected at the sensor nodes are iid, as training on iid data is likely to converge faster than training on non-iid data. However, this assumption is not applicable in FL.

A promising clustering FL approach has recently been examined in the literature to solve these challenges. WSN nodes in a clustering architecture, known as MNs, send their observations to their current CHs, which performs the learning process on the aggregated data at the local cluster level. Each CH then uploads its model parameters to the FL cloud server through the BS, where they are combined into a global model with the minimum possible frequency to reach convergence [83]. This clustering approach can help to reduce overall network energy consumption, as one aggregated transmission is much more energy-efficient than multiple separate transmissions, especially when the data size is large [15]. In addition, it can help reduce communication overhead, as data compression is possible in this approach [82]. A clear challenge with the clustering FL approach is the need to optimize the number of the CHs and the number of cluster members (CMs) per cluster, in addition to the possibility that a CH may fail to train or send its local model to the server.

The different approaches mentioned above share a common challenge related to the high number of transmissions required for the BS or the aggregator to broadcast the parameters of the developed model with the rest of the nodes in the WSN, which introduces a different communication overhead that requires high energy consumption [84,85].

8.3. ML Challenges in WSN

This section discusses challenges introduced by network resources, application and routing algorithms, the classical ML framework and, cross-layer attack detection when implementing ML techniques for the detection of cyberattacks targeting WSNs.

8.3.1. Challenges Related to Constrained Resources

ML algorithm selection should include consideration of the computational complexity, memory usage, and balance between the quality of learning and the associated energy budget, as the developed models are intended for deployment on resource-scarce devices. Continuous or periodic collection of network traffic results in big data issues, leading to a prominent challenge for the ML framework [86]. Moreover, the frequency of uploading data samples is different from one network scenario to another. For instance, certain networks are configured to put sensor nodes into deep sleep mode in order to conserve network energy; however, important readings may be missed in these scenarios, and a body of knowledge may be lost. Another example involves the possibility that network resource consumption may not be relative to the frequency of global aggregation and model training accuracy [87].

8.3.2. Challenges Related to Applications and Routing Algorithms

Developing a suitable ML security model to detect attacks for diverse WSN applications is challenging, especially for mission-critical, highly sensitive, real-time, and adversarial environment applications. It is preferable that anomaly detection be performed locally

at the local sensor nodes to avoid any communication with other nodes, the BS, or the IoT cloud due to high security requirements, which are not feasible for resource-scarce nodes. Another cyberattack detection challenge is the attacker's ability to exploit the routing algorithm and compromise its individual forwarding steps to attack the network. The purpose of these actions is to disrupt the routing and communication process by misdirecting or alternating the routing information or broadcasting fake information. The IDSs, on the other hand, can take advantage of the known behavior associated with the routing algorithm to build models of legitimate operation and compare them to the real exchange of routing messages between the nodes. Routing attacks belong to the network layer, and include sybil, 'Hello' flooding, sinkhole, blackhole, grayhole, and wormhole attacks [88,89]. Using secure routing protocols as a prevention technique and deploying proper ML-based IDS should be considered when securing these networks.

8.3.3. Challenges Related to the ML Framework

Pre-processing, feature selection and extraction, and hyperparameter tuning are essential for the success of any ML model learning process; however, collecting labeled data is not always possible in WSNs, as certain attacks may only appear in very few nodes and with low frequency. Thus, selecting an algorithm that can use minimally labeled data in a way that is sufficient for the learning process is crucial. Data reduction is required to reduce the processing time of the learning proccess on large datasets, especially for large-scale WSNs. ML preprocessing includes the process of adjusting the raw data to a format that can be used to train an ML model, such as removing features, sample size reduction, class rebalancing, missing data imputation, data normalization, encoding labeled data, and changing the data type of certain features. The process of reducing redundancy and correlation by selecting the most informative features during feature extraction while dropping irrelevant or partially relevant features from the dataset can be classified as follows: filter-based, wrapper-based, or embedded-based. Filter-based methods filter out irrelevant features independently of the learning algorithm, making it much faster and computationally effective than other methods and more suitable for WSNs [90]. Stacked-based feature extraction has been used as well; it combines several feature selection algorithms ordered as a stack and executed one after another, then applied to the dataset [91].

Hyperparameters are the set of parameters or arguments that are set manually before training and optimizing the ML model structure for better classification. These parameter value ranges are different for each ML algorithm. Hyperparameter selection significantly affects prediction results. Default parameters are the initial values that are pre-established when no values are explicitly provided. Optimized hyperparameters can be determined manually or automatically. Manual hyperparameter tuning is time-consuming, especially with a high number of possible combinations. Optimization algorithms can automate the process of finding the hyperparameters' this is called hyperparameter optimization. Different approaches include Bayesian optimization, grid search, random search, genetic algorithms, and particle swarm optimization. Several parameter combinations can be identified via search to determine the set of parameters that provide better detection results. Hyperparameter tuning is time-consuming when additional hyperparameters are added, as the number of parameter search combinations increases.

8.3.4. Challenges Related to Cross-Layer Attack Detection

Most of the existing techniques only mitigate specific types of attacks belonging to a single stack layer, excluding attacks on other layers. For instance, the network layer IDS can only detect routing attacks, and cannot recognize attacks belonging to the MAC, physical, or transport layers. It is essential to develop a cross-layer IDS that can detect different possible attacks that may occur at different WSN layers. Attacks can be identified by exploiting the information across the different layers to correlate the cross-layer features among them, such as between the MAC and network layers [86,88,89].

8.4. Datasets

Datasets are needed during the learning process to train and test ML models; therefore, dataset reliability and size are crucial to obtaining accurate results [92]. For instance, datasets that are large enough to have samples of normal traffic allow ML algorithms to learn normal network behavior, enabling the detection of unknown attacks by considering any deviation beyond the known usual behavior as unusual. It is challenging for the system to differentiate between the characteristics or signatures of a specific intrusion and a malfunction, which may cause false positives. Overall, any dataset collection for a specific network scenario should be performed over a sufficient time period to collect a sufficient volume of samples for each data class. Dataset parameters such as whether the dataset is balanced or imbalanced, the number of samples per class, dataset size, and dataset dimensionality can influence the selection of a proper ML classification technique and affect the behavior of the ML classifier.

Dataset quality affect the performance of ML models. First, ML algorithms used with a certain dataset may not be applicable for other datasets, as they may have differences in the number of classes to be distinguished, number of instances or samples for each class, and number of attributes that differentiate each class. Second, dataset characteristics such as being labeled or not (i.e., balanced or imbalanced), the number of features (i.e., dataset dimensionality), and feature importance can affect model quality. Feature extraction methods are usually used to filter out potential and relevant features. Third, the criticality and real-time nature of the WSN application at the time of data collection may result in noisy samples and irrelevant features, which can affect the final classification results and the ability of the trained model to differentiate between normal and abnormal behavior. For instance, increasing attack timespan traces and capture size can be used to control the imbalance within the dataset, thereby enhancing the learning process and allowing the algorithm to learn more differences between normal and attack samples. In addition, retraining ML models is possible, and can take place periodically during network progress as new or unknown attacks occur, allowing an ML model to modify its behavior and improve its detection accuracy.

Specialized datasets that consider the long list of cyberattacks targeting WSNs, whether collected using real-time experiments or computer-simulated, are limited. WSN-DS [92], NSL-KDD and its predecessor KDD Cup 1999 [93], CICIDS2017 [94], and UNSW-NB15 [95] are the most commonly used datasets been utilized for training and testing ML-based detection models in the context of securing WSNs. It is worth mentioning that none of these datasets except for WSN-DS are tailored to the need of developing ML models for WSNs, which motivates a need to generate new datasets or collect actual logs of real normal network data and simulated attacks.

8.5. Evaluation Metrics

Two types of cyberattack classifications are present in the literature, based on the number of classes (i.e., attacks): binary classification, in which there are only two classes, attack or normal; and multi-class classification, in which the number of considered classes is greater than two if more than one attack has been detected and sampled in the dataset. In both cases, the testing phase in the process of developing a ML model involves different evaluation metrics, which can include P_d, P_{fa}, P_{md}, positive prediction value (PPV), ACC, Error rate (ERR), geometric Mean (GM), root mean square error (RMSE), normalized RMSE, normalized RMSE (NRMSE), receiver operating characteristics (ROC), and $F1 - score$, which can be expressed as follows:

$$P_d = \frac{T_P}{T_P + F_N}$$
$$P_{fa} = \frac{F_P}{T_F + F_N}$$
$$P_{md} = \frac{F_N}{T_N + F_P}$$
$$PPV = \frac{T_P}{T_P + F_P}$$
$$ACC = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}$$
$$GM = \sqrt{(P_d * P_{md})} = \sqrt{(T_p/(T_p + F_P) * T_N/(T_N + F_N))}$$
$$ERR = (1 - ACC) = \frac{F_P + F_N}{T_P + T_N + F_P + F_N}$$
$$F1 - score = \frac{2(P_d * PPV)}{(P_d + PPV)}$$

where T_P, T_N, F_P, and F_N are the number of true positives, true negatives, false positives, and false negatives, respectively, as per the confusion matrix illustrated in Figure 7.

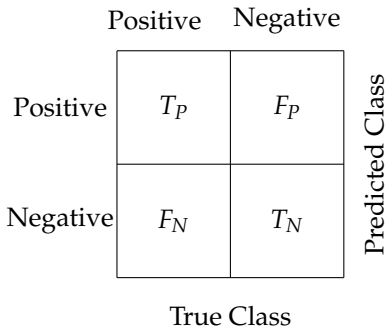

Figure 7. Confusion matrix.

These attributes are estimated after dataset testing and are calculated from the generated confusion matrix.

P_d, called the sensitivity, recall, and detection rate or true positive rate, corresponds to the number of correctly detected attacks vs. the total number of attacks. P_{fa}, or the false alarm rate, corresponds to the number of incorrectly detected attacks vs. the total number of normal traffic instances. P_{md}, or the false negative rate, is the number of undetected attacks vs. the total number of normal traffic instances. ACC is the measure of correctly detected traffic instances, whether normal or attack, vs. the total number of detected samples. ERR is the complement of ACC; it is the misclassification rate, which provides a measure of incorrectly detected traffic instances vs. the total number of detected samples. PPV represents the total number of correctly detected attacks vs. the total number of correctly and incorrectly detected attacks [96]. The F1-score or F-measure represents the harmonic mean of precision and recall; it uses F_N and F_P to efficiently classify noisy or imbalanced data [97]. High ACC, P_d, PPV, F1-score, and GM and low P_{fa} and P_{md} values generally indicate that an ML model has the potential to accurately detect attacks while ensuring that a low number of attacks go undetected. In addition, RMSE and NRMSE are used to evaluate different cyberattack detection methods numerically, and can be expressed mathematically as

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(A_i - \hat{A}_i)}{N}} * 100\%$$

where i is the index of the evaluated sample, A_i is the actual value, $\hat{A}_i$ is the estimated or predicted value, and N is the total number of tested samples. NRMSE is defined as a measure of a model's predictive power against simple prediction using the mean of the observed data, and facilitates comparison between models with different scales; it is calculated as follows:

$$\text{NRMSE} = \frac{\text{RMSE}}{A}$$

where A represents the mean of the observed data values, which can be replaced with a range defined as the difference between the maximum and the minimum values of the observed samples. The ROC curve plot indicates the tradeoff between P_d (the Y-axis) and P_{md} (the X-axis). Preferably, the area under the ROC curve should be close to unity; low values are an indication of weak model performance in terms of detection [98]. NRMSE can be interpreted as a fraction of the overall range that is typically resolved by the model. A lower RMSE is preferable. This value is minimized when the predicted value matches the true observed value from the environment.

Evaluation metrics such as PPV, ACC, ERR, and F1-score are computed using values from the confusion matrix in both columns, and as such are sensitive to any change, especially with an imbalanced dataset. These metrics change as the distribution of data changes, even if the classifier's performance does not [96]. However, GM can be used with both balanced and imbalanced data, even if its calculation involves values from both columns of the confusion matrix, because the changes in the class distribution cancel each other out.

Other evaluation metrics commonly used to assess the ability of ML models to detect cyberattacks targeting WSNs are related to the required memory usage, buffer size, computational complexity, processing time, and prediction time, which are other elementary evaluation metrics. On-chip memory usage considers the random access memory (RAM) and Flash memory in the microcontroller unit, usually measured in kilobytes (KB). The amount of RAM directly affects processing speed. A larger amount can handle more data; however, WSN nodes have relatively low on-chip memory, which means that ML models must require low amounts of on-chip memory and be optimized for efficiency. Buffer size affects the rate of false alarms, as the node buffer usually stores certain fields of monitored traffic data that can be used as input for the detection model running within the nodes. In certain scenarios, a specific MN is responsible for monitoring its neighbors, listening to messages within radio range, and continuously examining traffic to look for intruders.

9. BC and Cyberattack Prevention

One of the earliest data security techniques of major significance is digital time-stamping, which was proposed by the authors of [99] in 1991 and has drawn the attention of industry and academia ever since. The work in [99] proposed using a family of cryptographically secure collision-free hash functions, digital signatures, and linking schemes to preserve the sequential occurrence of the client's requests in the network. Digital time-stamping is the precursor of the well-known BC technique [100], which is discussed in the following subsections.

9.1. BC Background

A BC is a set of blocks, with each block being a combination of an individual set of transactions. The number of transactions in each block depends on the block size and the transaction size. The blocks are linked using cryptographical sequential digital signatures [101,102]. These signatures are chained utilizing a hash value that involves data from the previous and current blocks to preserve the authenticity of the block's content against any data tampering [100,103]. The chain starts with a genesis block, which is the first block in the chain [104], and each subsequent block is added based on a distributed consensus with a hash value and timestamp. The shared ledger in each node connected to the BC network is updated through a consensus algorithm with each added block. The

consensus mechanism ensures a common ledger database that is difficult to tamper with and has unified content on all nodes.

Figure 8 depicts the general structure of the block. Each block consists of a block header and block body. The block header contains that block's metadata, including its version number, previous block hash, nonce, Merkle root, timestamp, and nBits, while the block body consists of the transactions embedded in the block [105]. An explanation of each component is provided below.

- Version number: indicates validation rules that the BC must follow.
- Previous hash: a 256-bit value that points to the previous block (sometimes called a parent block) and affects the current block's hash to ensure the chain structure's uniqueness.
- Timestamp: the block's approximate creation time, which is required for traceability.
- Nonce: a one-time use number in the block header that is required in order to state the number of leading zeros for the hash value. This number can then be used to determine the level of difficulty when calculating the hash of a block and for verification to ensure consensus.
- Merkle root: sometimes called the 'hash of all hashes', this uniquely identifies the block; its calculation depends on the block's transactions [106]. The Merkle tree is used to verify the validity of the transactions instead of downloading the entire chain. Figure 8 illustrates the Merkle tree's structure, which is represented through the individual hashes of the transactions or leaf nodes; each set of child hashes is combined and hashed again up the tree until the root is reached [102]. Changing one transaction causes a change in the whole chain of hashes up to the Merkle root value [102].
- Hash target or nBits: a threshold value that the block header hash must not exceed in order for the block to be valid; the nBit value is usually continuously adjustable and increases with the number of leading zeros.

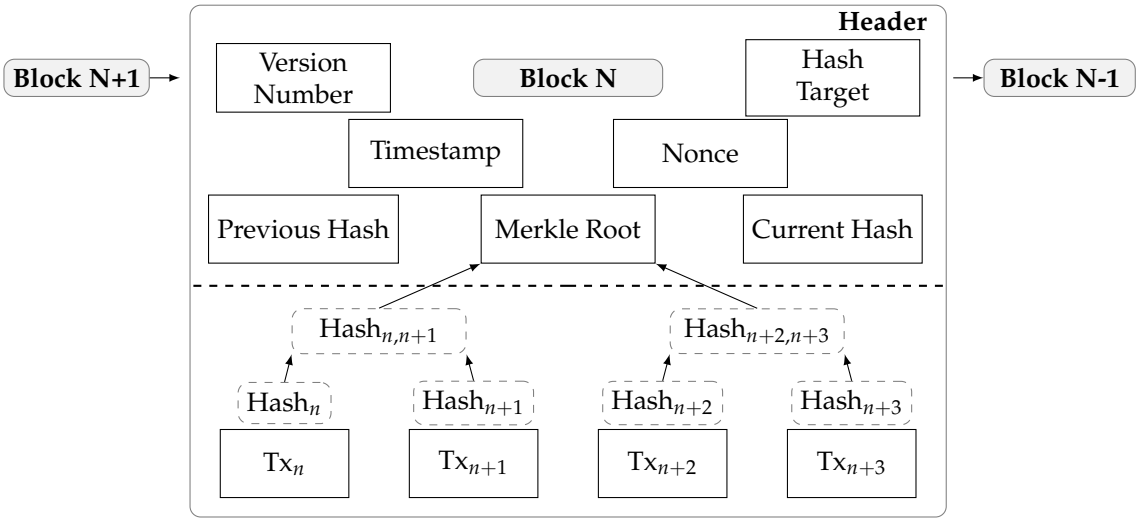

Figure 8. Block structure.

BC eliminates the need for a third-party central authority, as it is distributed or decentralized and comprises all committed transactions in the network; this makes it useful for securing cryptocurrency systems. In addition, BC has an ideal architecture for many applications that require ensuring distributed transactions between nodes and decentralizing computation and management in a trustless environment [107]. Using BC in IoT systems can reduce security risks by safely storing data, routing, accessing resources, and authenticating identities [108]. As discussed in [109], BC is a promising approach for securing data and authenticating identities in IoT because of its peer-to-peer (P2P)

distributed ledger, which supports scalability and faster settlement for coordinating and securing joining nodes. However, the challenge of applying BC in WSN is its high demand in terms of storage and computational complexity, which causes additional delays and reduces network throughput. BC is often costly in terms of communication, memory usage, and power consumption, while sensor devices are typically designed to be low-cost with restricted resources; however, the cost of setting up and maintaining a centralized database can be reduced with BC. A node's idle state can be fully utilized in terms of the device's computational, storage, and bandwidth capabilities, thereby lowering overall network calculation and storage costs.

Overall, using BC to secure a WSN has many advantages; however, it is difficult to develop lightweight BC security mechanisms that carefully consider the tradeoffs between BC security and WSN design factors in terms of power and latency [110]. This work aims to investigate how BC can effectively protect sensor nodes from possible cyberattacks and determine its appropriateness for WSN applications.

9.2. BC Features

The main keywords or features that describe BC are illustrated below.

- Data immutability: data are protected using cryptographic hashes unique to each block, disallowing manipulation or alteration after registration in the BC network.
- Decentralization: the absence of a trusted supervised centralized authority; decentralization ensures a lower failure rate, makes the network less prone to malicious attacks, and reduces reliance on a third party.
- Transparency: every involved node in the network is aware of the updated stored data.
- Security and Resilience: any data manipulation requires the approval of more than half of the miner nodes, which is extremely difficult to obtain practically.
- Data Encryption: the provision of public and private keys for data encryption and decryption, respectively, via the use of an asymmetric encryption algorithm for every two communicating nodes; the public key is shared between all nodes in the network to encrypt the data, and the targeted receiver can decrypt the data using its own private key.
- Digital Signatures: digital signing of transactions using a digital signature algorithm, such as the elliptic-curve digital signature algorithm (ECDSA), to approve transaction content and originate node identities.
- Consensus: every node in the network should agree on the current state of the distributed ledger, which is made possible using one of several popular consensus mechanisms, such as Proof-of-Work (PoW),Proof of Authority (PoA), Proof of Capacity (PoC),Proof of Share Stake (PoS), Delegated Proof of Stake (DPoS), Raft, Proof of Elapsed Time (PoET), and practical Byzantine fault tolerance (PBFT) [111].
- Smart contract: a piece of code that adds customizability to a BC. It represents an arrangement and executes itself automatically under a predetermined set of rules and conditions without a third party. Smart contracts can be used for node verification and authentication. The input of the smart contract is the transaction, which is executed with a corresponding code that consists of the value, address, functions, and state to generate the output events (see Figure 9) [112].

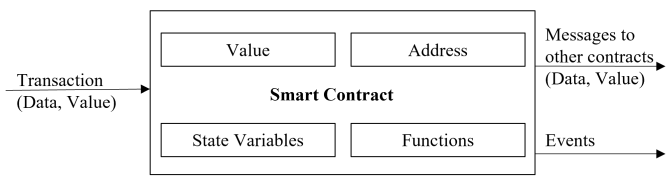

Figure 9. Smart contract structure.

9.3. Types of BC

There are four primary types of BC platforms: public (or permissionless), private (or permissioned), consortium (or federated), and hybrid (Figure 10). A BC is a fully or partially decentralized architecture that authenticates sensor devices joining the network and accepts or rejects transactions. A public BC is completely distributed; it allows any node to join the BC with similar access rights, generate new blocks, and validate data blocks. Public access to the BC provides data availability, transparency, and confidentiality. Examples of public BC platforms include Ethereum and Kadena [113]. A private BC has a central authority (or network manager) that determines which nodes may join, and does not provide each node with equal rights to perform tasks [114]. It differs from a public BC in that it restricts node participation and access to the BC depending on the authorization provided by the network [115]. Examples of private BC platform include Hyperledger Fabric, Hyperledger Burrow, IOTA, Quorum, Corda, Tendermint, Symbiont, HydraChain, Exonum, and Multichain. Both types, private and public, have disadvantages; for instance, public BCs tend to have a longer validation period for new data than private BCs, while private BCs are more exposed to certain types of cyberattacks. Compared to public BCs, several research works have considered private BCs to be advantageous when used in IoT systems, particularly in terms of network latency, due to the additional time required by public BCs to obtain consensus between all peers. PA private BC is fully controlled by one organization, and trust comes from preselecting which nodes are authorized to use the shared ledger and to verify transactions; as there are fewer trust difficulties, fewer security measures are necessary between nodes, creating a more responsive network, which is needed in IoT deployments in terms of scalability. Another advantage of private BC implementation is higher data privacy, as network data are limited to the private network and are entirely controlled by the network manager. Changes can only be made by certain nodes within the network, though all network nodes can read the data within the private BC. The role of network miners, called voters, validators, or peers, is to approve transactions and maintain copies of the BC ledger, which helps to secure and stabilize the private BC network. Because only a few nodes are delegated to publish blocks within the network in a private BC, they are more vulnerable to certain attacks types, as the authority may modify or tamper with rules or even data, and the organization may choose to revoke their BC to a previous time instant [116].

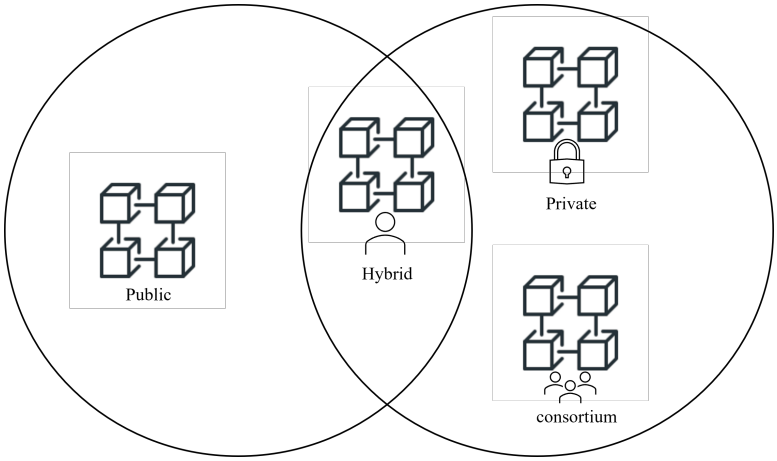

Figure 10. Types of BC.

When using a private BC for an IoT application, all nodes are identified before deployment using a Certificate Authority (CA) or Membership Services Provider, which releases identities, or key pairs, for IoT nodes. Each IoT node's registration is performed on the BC using its cryptographic hardware identity hash. Node registration is performed by mapping an IoT device's public keys and their identities, which must be stored on the BC [24]. This stops the BC from receiving information from unauthorized IoT nodes,

securing it against potential attacks. A consortium BC is a type of permissionless BC; it is partially decentralized, as it is governed by a group of preselected nodes that directly participate in the consensus mechanism, instead of a single central entity as in a private BC. A consortium BC is more decentralized than a private BC, and provides better security; however, establishing a consortium requires cooperation between a number of key nodes (sometimes called organizations), which presents logistical challenges and increases potential risk in cases where a majority of the consortium wants to tamper with the BC. A hybrid BC refers to a customizable BC architecture that combines features of both private and public BCs. Hybrid BCs are best suited for systems that cannot be fully private or public and involve a lack of trust, such as IoT, supply chains, finance, and banking.

9.4. Performance Evaluation Metrics

The performance of a BC security system used in a WSN depends on the effectiveness of its peer trust, node authentication, access control, smart contracts, consensus mechanisms, resources management, and big data processing and storage. There are multiple performance criteria of interest, including transaction throughput, response time, latency, storage overhead, and energy consumption, which are the most commonly used metrics for security analysis of BC-based WSNs, and can be defined as follows [117]:

- Transaction Throughput: the maximum amount of transactions that are processed and committed by the BC in a specific period of time, usually represented by transactions per second (tps).
- Response time: the time required to handle and verify the transactions processed by peers. The response time increases with increasing batch size, such as when the number of transactions in the queue grows; this can result in system congestion, as peers are required to handle more transactions.
- Latency: the period of time between when the transaction is invoked by a node and the time the transaction is added to the ledger.
- Transaction size: the amount of data in the transaction to be added to the next block.
- Block size: the size of the block, that is, the number of transactions included in the block.
- Storage Overhead: the storage capacity required for BC operations, which may exceed the node's storage capabilities due to the large amount of data accumulated by security tasks [118].
- Residual energy: the remaining energy in the sensor nodes; this metric is important to consider for energy-related attacks which shorten the network lifetime by wasting nodes' energy by launching malicious activities [119].

Other metrics include processing frequency, percentage of central processing unit (CPU) usage, computational complexity of encryption [120], and processing time of trust evaluation. In addition, the authors of [121] used the probability of attack success and strength of attack detection to evaluate secure mechanisms using BC. The probability of attack detection identifies how efficiently a secured mechanism can distinguish between legitimate and malicious entities targeting an IoT network, while attack strength is determined by an attacker's ability to compromise a certain node and force the network to behave maliciously.

9.5. Securing WSNs Using BC

A typical BC procedure in a P2P WSN begins when a transaction is launched between two sensor nodes. This transaction is hashed and broadcast to the P2P network. The nodes involved in the interaction sign the transaction using their public keys, as several nodes are involved in the forwarding path in a multi-path forwarding scenario. The transactions are verified and validated based on the consensus mechanism in terms of data and identity by miners or voters, then disseminated, stored, and grouped into a block. The new block is sent to the BC P2P network to be added to the chain when complete. The chain is shared, immutable, and tamper-proof across the participant nodes (Figure 11).

Two architectures are most common to build BC-based security systems in WSN contexts, namely, centralized (Figure 12) and cluster-based (Figure 13) [122]. In addition, there are two types of nodes, full and lightweight[123]. Full or Aggregator nodes store the complete ledger locally; therefore, they have access to the complete transaction history in the chain. In a WSN, a full node is usually a BS or CH. Lightweight nodes do not store a complete ledger; they only store the BC transactions with high importance and what is relevant to their operation, placing their "trust" in their associated full nodes. These nodes are usually the terminal nodes or CM. In this way, the download and storage requirements of these nodes are reduced. These architectures align with private BCs; however, it is not recommended to have a single central node similar to a BC or a CH as a master authority in charge of authentication and trust management in order to avoid any critical points of vulnerability in the network.

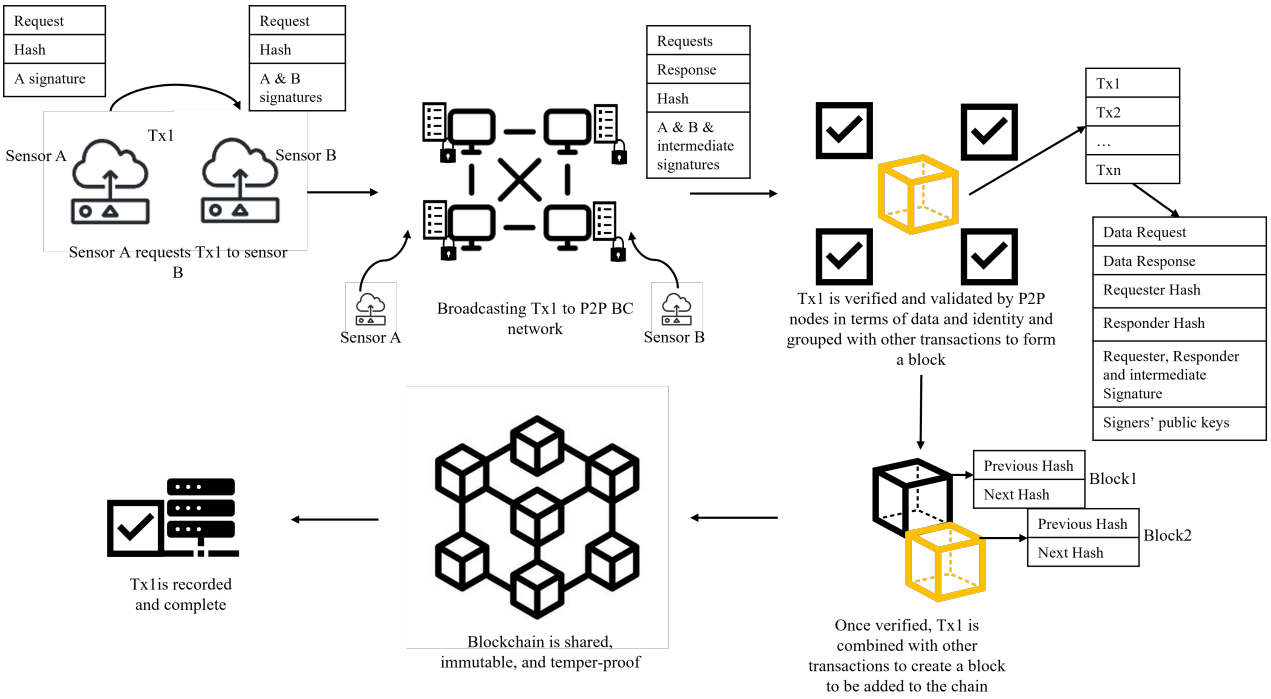

Figure 11. BC workflow in WSN.

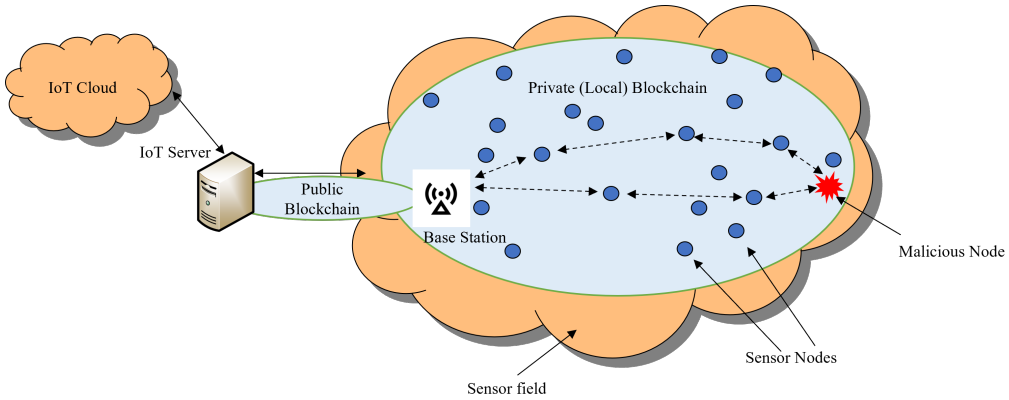

Figure 12. Centralized architecture in a BC-based WSN.

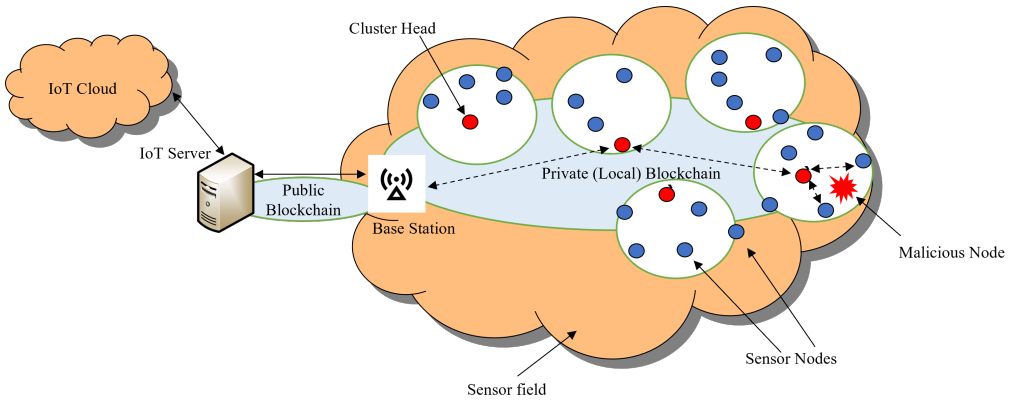

Figure 13. Cluster-based architecture in a BC-based WSN.

9.6. *BC Challenges in WSNs*

- Scalability and Storage: the amount of generated data grows exponentially with an increase in the number of devices deployed in different IoT applications, which leads to an increase in the rate of transaction execution and in the storage capacity needed to keep the ledger up to date [124]. Scalability becomes a severe bottleneck with an increasing number of transactions, and limits the practical development of BC in WSN contexts. With BC technology, blocks are not stored in a central server; however, a subset of the nodes need to keep a copy of the entire ledger in their own limited storage, which means that maintaining enough storage space for the ledger may not be feasible. Moreover, the size of the ledger increases over time, while most of nodes have low storage capacity of 10 KB to 100 KB memory at the most.
  The ledger storage requirement remains an open research issue. According to [125], certain IoT devices are limited to up to 8 MB of memory, most of which is used for storing the software that manages the device; therefore, lightweight mechanisms that limit the ledger size and allow it to be stored by each node are highly recommended. The authors of [118] defined three strategies for data storage by IoT sensors: full storage, in which all nodes store the full current data; partial storage, in which each node stores only part of the data, allowing it to be restored when combined with data from other nodes; and persistent storage, in which low-priority or old data can be stored in a remote centralized database. Similar criteria can be suggested to reduce storage overhead.
  Efficient consensus protocols, optimizing block size, sharding, pruning, lighting protocols, and off-chain storage have been proposed in the literature to address scalability issues. For instance, PBFT is considered a suitable protocol for fixed and small-size networks, although it is not scalable for larger numbers of IoT devices [126]. Sharding is one of the newer mechanisms to support scalability; it aims to split the overhead of processing transactions between multiple 'shards', or subgroups, of consensus nodes. These groups work in parallel to maximize performance while significantly avoiding the overhead due to duplication of communication, computation, and storage per full node, allowing the system to scale to larger networks [127]. Scalability can be increased by pruning the size of blocks on the BC, which includes removing older transactions to control memory usage [128]. Lighting protocols aim to lower the verification process period by only allowing full nodes to store the complete ledger, with lightweight nodes only keeping a portion of it. In off-chain storage, only hashes are stored in the ledger, whereas actual data are stored off-chain i.e., in the cloud, to support the scalability in dense WSNs.
- Consensus mechanisms: common consensus mechanisms such as PoA, PoS, and PoC are primarily designed to work for monetary transactions, and are not suitable for adoption in WSNs and constrained-resource IoT devices [129]. PoW is not common in IoT and WSN applications, as it is demanding in terms of computational power;

while PoC is energy efficient, it depends on a node's storage capacity and monetary stake, and monetary stakes do not exist in IoT and WSNs [130]. In addition, the most commonly known consensus mechanisms do not perform as desired in their raw mode because of massive requirements and scalability issues [131].

- Communication Overhead and Synchronization: a significant amount of communication overhead is required to synchronize the BC copies, as there is a need to forward every verified transaction to all peers. Establishing keys and authenticating nodes with cryptography, which is determined by the encryption type (either asymmetric or symmetric), causes high communication overhead and key storage [118]. Time consistency during time synchronization between sensor nodes requires exchanging a number of messages depending on synchronization frequency.
- Computation Overhead: heterogeneous IoT devices have different processing capabilities for running encryption and decryption, which leads to variations in processing time. Integrating a BC into the sensor network enables a logical peer-to-peer network to validate and store transactions locally, which is straightforward for personal computers or workstations; however, it might be difficult for tiny sensors with limited computational resources.
- Complexity and Energy Wastage: most widely employed BCs use PoW as a consensus mechanism, where the network participants must solve a mathematical problem or cryptographic puzzle in order to validate and authenticate transactions. PoW uses a significant amount of computational resources, causing energy losses; therefore, it is not practically suitable for IoT networks [30,111]. In addition, severe latency affects WSN stability in delay-sensitive applications [132].
- Guaranteeing Security: many malicious activities target IoT and WSNs. A single attack can harm a large number of devices or be used to destroy another system, as monitor nodes can be turned into malicious nodes to launch further attacks. The network's ability to manage advanced cyberattacks is degraded due to the constrained resources of IoT devices. However, a BC relies on sophisticated hash functions, which require heavy computation and consensus mechanisms that consume network bandwidth.
- Compatibility and Standardization: standardization for BC security applications is needed in to ensure that devices meet a reasonable set of standards and have fundamental security and privacy capabilities and to diminish risks associated with cyberattacks against IoT devices [133].

BC in WSN and IoT may never become a reality unless the storage, battery life, computation power, and bandwidth availability of sensor devices are improved [134]. Securing a network using a BC with resource-constrained devices remains challenging [111], and researchers are currently seeking lightweight mechanisms that can solve problem of excessive resources consumption by sensor networks when using BC to secure WSN and IoT networks against possible attacks such as data manipulation and tampering.

9.7. BC in the Literature

BC is a dynamic technology that has spurred tremendous technological advances in many fields in the last few years. BC has been recently proposed as a method to secure the systems applications associated with IoT, such as smart homes, supply chains, smart agriculture, and smart grids [135]. The recent focus in the literature has shifted to securing WSNs using BC, despite the challenging characteristics and performance limits discussed in Section 9.6. BC characteristics, which drive many WSN design challenges include ledger size, block size, number of transactions per block, smart contracts, miner selection criteria, number of miners, selected BC type, and hash function specifications. Using BC to secure WSNs includes proposing mechanisms to protect data sharing, establish trust, authenticate identities, secure routing tables, and secure localization against dangerous cyberattacks such as sybil, spoofing, DoS, message substitution, and replay attacks (see Table 3). The rest of this section examines BC techniques presented in the literature while highlighting their advantages and drawbacks.

Malicious node detection is one of the main applications of BC in the context of WSNs [31], and is the focus of [136], where the authors studied the use of BC in conjunction with smart contracts to identify malicious nodes. The authors proposed a trust model using smart contracts based on the processing delay, forwarding rate, and response time as evaluation metrics to distinguish malicious and benign nodes. The result of the detection process was then recorded in the BC. The work in [136] established that the proposed method is effective in terms of detecting malicious nodes and allows detection process traceability; however, the adopted traditional consensus method, PoW, is computationally demanding and is unsuitable for resource-limited WSN nodes.

Another major application is maintaining data authenticity, which is ensured using node authentication and trust management [52,106,109,136–143]. Trust management is tied to authentication mechanisms, which identify end-communicating nodes and ensure data validity and confidentiality. The authors of [138] proposed a BC-based trust model and node authentication using a smart contract at gateway nodes such as CHs and BS to reduce energy consumption, claiming that the model takes 0.000250 s, unlike Ethereum BC, which requires 14 s to achieve the same results. The benefits of this type of model were further discussed in [140] through a test-bed experiment, which was conducted to determine whether a BC-based data-driven trust mechanism could reduce network transaction throughput in the presence of grayhole and blackhole attacks. The authors of [143] examined ways of reducing the communication overhead associated with BC by reducing the size of public and private keys. This reduction was achieved in [139] by employing Hyperel-liptic Curve Cryptography (HECC), which can potentially provide a similar security level to other key generators along with a lower key size. Another practical implementation of integrating BC into a WSN to protect data against tampering was discussed and evaluated in [52,106,141,142]. The performance evaluation in [106] indicated that the computational complexity associated with evaluating the hash function increased as the amount of sensed data increased, as the ledger size increased accordingly, resulting in reduced data transfer efficiency in the network. The work of [52] proposed limiting the size of each BC and setting a time window with a circular buffer mechanism to minimize BC length as a solution to this limitation. The authors of [52] considered a hash and an additional time interval measure to determine nodes' reliability levels, which is equivalent to the dynamic value of the accumulated trust points of a certain node. This reliability level was controlled (increased or decreased) by tracking a ledger-checking message. The experimental results of [52] indicated that the proposed trust mechanism could reduce both communication overhead and memory requirements. A different approach to protecting data integrity was proposed in [120], where the authors focused on using a cryptographic algorithm in conjunction with a private BC to protect data during transmission between nodes. The authors proposed a methodology integrating BC and Advanced Encryption Standard (AES) symmetric encryption in a WSN system, with the hash value used to encrypt data during the transaction and AES used in the data transport layer as an encryption/decryption process carried out between any two communicating nodes. This methodology reduces resources consumption while protecting the network against linking, MITM, and Distributed denial of service (DDoS) attacks. The method proposed in [120] is not scalable, however, as it is limited to the use of private BCs. The authors of [129] proposed and simulated a new model for a consensus algorithm that can reduce the time required for mining. The results presented in [129] confirmed that the proposed model can potentially protect the network against spoofing and injecting phantom devices; however, the software associated with the model cannot be updated.

Protecting the network routing process using a BC was the focus of the work in [141,144,145]. The authors of [144] proposed a BC-based routing protocol that uses a BC to store the network's activities and broadcast the status of the nodes. In [144], the aim of the authors was to secure the route determination process by avoiding untrusted nodes and to resolve the load-balancing issues associated with routing. The authors of [141] proposed using the a BC-block approach with flow routing tables instead of converting the entire SDN-enabled

WSN network to a BC network, thereby preventing tampering with flow entries. This approach reduces the energy consumption associated with traditional BC algorithms; the results obtained in [141] using a simulated Riverbed model indicate that the proposed scheme can provide security against MITM, replay, and blackhole attacks, though with increased energy consumption and end-to-end delay. A different distributed ledger-based technology was considered in [142] as an effective lightweight network to authenticate and protect routing tables against sybil attacks in addition to protecting the network against fake identities more broadly. However, the proposed network in [142] is time-consuming, not scalable, and has a centralized architecture. The authors of [145] proposed a trust model for a decentralized architecture to secure WSN routing through a dual BC model. The first model is public and implements a PoW consensus to authenticate aggregating nodes (ANs), while the second model is private and uses PoA for authenticated sensor nodes. The PoW mechanism enhances the security level of the unauthenticated public BC, which includes the BSs, though at the cost of high computational complexity. On the other hand, PoA is less computationally complex, helping to reduce the overhead of the resource-limited AN, which is included in the authenticated private BC. Node integrity is evaluated using a trust evaluation metric to determine the legitimate nodes that take part in the routing process; however, the security analysis in [145] indicated that the proposed approach could be vulnerable to smart contract-based attacks resulting from possible bugs in smart contract code, such as integer underflow, overflow, parity multisig, timestamp dependency, transaction ordering dependence, call stack depth attack, and re-entrancy.

The authors of [109,139,146] proposed BC-based identity authentication mechanisms for sensor nodes joining the network. The authors of [109] proposed a secure identity authentication mechanism in a hierarchical architecture for a multi-WSN environment using public and private BC, where the former includes the BS and terminal users as miners, while the latter is composed of all authenticated CHs. This technique minimizes communication overhead, as sensing nodes are not connected directly to the unauthenticated public BC; therefore, frequent node authentication is not required. The focus of [146] was on securing an identity authentication scheme against worm attacks by using the IOTA Tangle BC to store the authentication data safely; however, the network proposed in [146] relies on a single point, namely, sink nodes, to authenticate other nodes, which means that the network has a centralized architecture. Another approach was proposed in [147], where the authors considered a sequential detection scheme that starts by validating the hash value of the node's ID, followed by validation of the node signature by each node, and ends with a voting technique that determines whether the node is malicious or benign. The results from the different stages are then used to decide whether a suspect node is kept or eliminated. The authors of [147] revealed that the security level of the proposed BC method is improved compared to other classical approaches in the literature; however, the latency introduced by the three combined phases could potentially be higher than classical approaches. The authors of [148] proposed another decentralized authentication and trust model, which stores the authentication and trust information in the BC and uses a subjective probability as a reputation level. The technique is limited by the origin block problem, which causes the system to misbehave in cases where malicious values are included in the first block in the chain.

Localizing WSN nodes accurately is another application, and was addressed in [149] by investigating a decentralized BC-based trust management model. Their model relied on a trust value consisting of both behavior and data trust values evaluated by a selected number of trusted nodes, such as the number of successful and unsuccessful interactions between sensor nodes and feedback metrics related to the integrity of each beacon node. Though the simulated results in [149] indicate that the proposed scheme outperforms other current techniques in several aspects, it requires an additional number of transactions associated with the evaluation processes, and lacks a complexity analysis of the proposed technique.

Table 3. Existing research on BC-based WSN security (N/A means not available).

BC Type	Ref.	Security Threat	Study Highlights
Private	[120]	DDOS and Linking attacks	Relies only on AES symmetric encryption for data integrity
	[138]	Data Tampering	BC-based trust model and node authentication using smart contracts to reduce latency.
	[106]	Data Tampering	Performance evaluation of the computational complexity associated with the ledger.
	[147]	Internal attacks	Three-phase sequential detection using sensor node hash values, node signatures, and voting degree
	[146]	Worm attack	Relies on IOTA Tangle
	[142]	Sybil attack	Relies on IOTA Tangle
Consortium	[136]	Internal attacks	Utilizes a smart contract in conjunction with BC
Hybrid	[109]	Sybil, MITM, DoS, Message Substitution, and Replay attacks	Identity management and secure authentication mechanism
	[139]	Internal attacks	Employs HECC to generate public and private keys
	[145]	DoS and Sybil attacks	Employs dual public and private BCs which implement a PoW and PoA consensus, respectively, for authentication of each BC.
N/A	[140]	Greyhole and blackhole internal attacks	Test-bed experiments using a data-driven trust mechanism to reduce network transaction throughput.
	[52]	Physical or logical data tampering	Limits the size of the BC and uses a time window with a circular buffer mechanism to reduce the BC's length.
	[149]	Attacks on the localization process	Uses both behaviour and data trust values to determine the reliability level.

Table 3. *Cont.*

BC Type	Ref.	Security Threat	Study Highlights
	[129]	Injection, data tampering, firmware modification, listening to traffic	Proposes a new consensus algorithm model to reduce mining time.
	[144]	Routing attacks	Utilizes BC as a shared memory for route determination to avoid untrusted nodes
	[141]	Routing attacks (blackhole, replay, and MITM)	Relies on BC block technology to protect the flow routing tables at the nodes against routing attacks in an SDN-enabled WSN
	[148]	Internal attacks	Uses a subjective probability measure as a reputation level model for peer trust

10. BC–ML Integration

It is evident from Sections 8 and 9 that BC technology has been found in the literature to be a useful framework for securely recording data transactions in a tamper-proof ledger with the help of embedded mechanisms such as consensus and smart contracts, whereas ML provides efficient classification models to identify attacks. Therefore, when considering integrated BC and ML approaches, BC technology can help to securely store data generated by WSN devices. This generates huge amounts of data, which can then be modified and organized to safely train ML classifiers, potentially achieving high detection accuracy. It is notable that the output of the ML detection process can be securely stored on the BC network to preserve the integrity of the detection process results. Figure 14 depicts key features of the integrated BC–ML security approach. Despite the potential benefits and the fact that their integration is possible, inevitable, and beneficial, integrating these two technologies simultaneously poses new challenges when adopted in any WSN application. Most of the existing literature has studied ML and BC separately when considering securing WSNs, unlike other IoT applications such as smart grids and supply chains. However, the gains to be achieved and the challenges faced when seeking to combine these two technologies for securing WSNs have not yet been extensively explored in the literature due to its being a relatively new research direction.

In this regard, our approach is to have two lines of defense utilizing the integration of BC and ML. The first line of defense is attack prevention using BC, while the second line of defense is attack detection using ML. In case the first line of defense fails to prevent an attack, the second should verify and examine the incoming traffic for any sign of vulnerability, alerting the network to the presence of a malicious attack [150].

The emphasis of this section is on those research works that consider BC–ML integration to secure WSNs, as discussed in Section 10.1. Following a discussion of the important open issues and research challenges involved in the interrelationship and integration of both technologies to protect WSNs against cyberattacks, which is detailed in Section 10.2, this section is closed by detailing our proposed approach to an integrated BC–ML solution.

10.1. Related Work

Integrating the technologies of ML and BC to secure WSNs has been considered in the literature in several different directions, namely, secure routing, secure authentication, malicious nodes detection, and trust mechanism, as outlined in Table 4.

Table 4. Existing research works on BC and ML integration for securing WSNs (N/A means not available).

BC	ML	Ref.	Attack	Study Highlights
Public	Deep CNN	[151]	Internal	Trusted distributed routing using BC while avoiding routing paths with congestion and malicious nodes using DL-CNN
	Hidden Markov Model (HMM)	[137]	Sybil	Trust model for identifying Sybil nodes; trust value is evaluated via HMM, and the trust values are then added to the BC
	Histogram Gradient Boost (HGB)	[152]	DoS	BC-based authentication mechanism in which IPFS is integrated with BC for data storage, and HGB detection module to mitigate DoS attacks
Private	Isolated forest algorithm	[153]	Internal	Isolated forest algorithm for anomaly detection in the BC
	Genetic Algorithm (GA)-based SVM and GA-based DT	[154]	Grayhole, mistreatment, and MITM	ML models for malicious node detection and registration along with authentication mechanism and data storage routing using BC
Consortium	RL	[155]	Blackhole	Trusted routing with the use of BC and reinforcement learning
	DNN	[156]	Routing, specifically Blackhole	Trusted routing with the use of BC and DNN
Hybrid	Gaussian NB	[122]	Internal	BC-based identity management and secure authentication mechanism with Gaussian NB detection module to mitigate DoS attacks
N/A	Generative Adversarial Networks (GAN)	[157]	Network layer	Authentication and validation of current routing data using a Generative Adversarial Network-based BC-enabled secured routing protocol

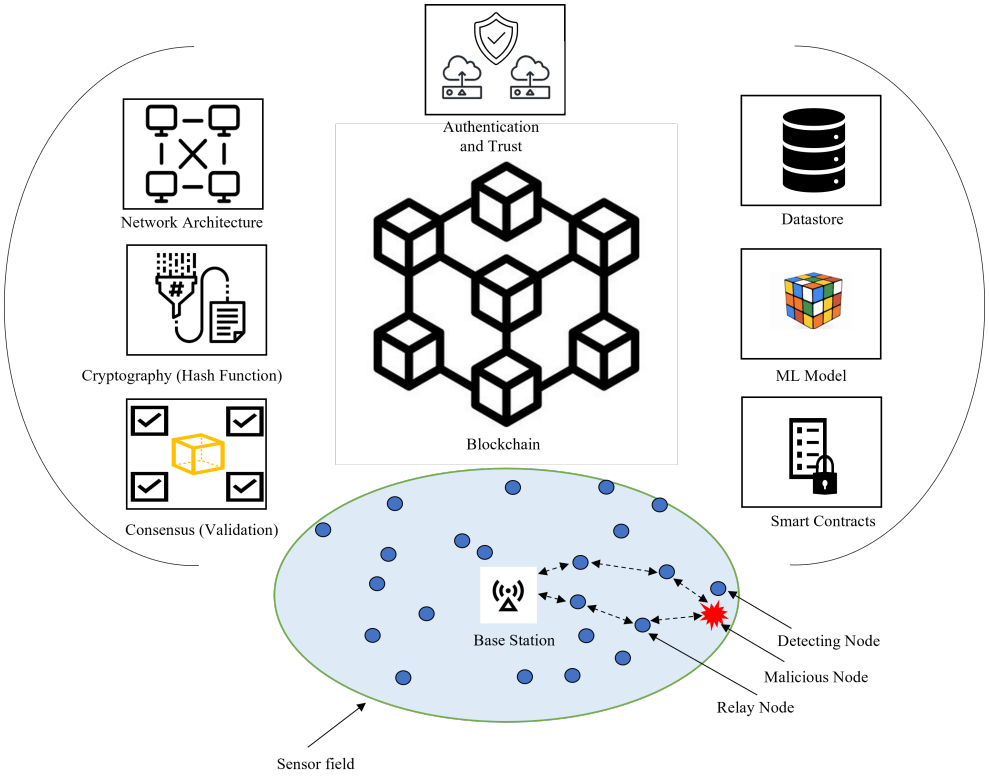

Figure 14. Key features of BC–ML integration for WSNs.

Securing WSN routing protocols using an integrated BC–ML approach has been considered in [151,155–157]. The proposed framework in [155] relies on the BC network to securely record the routing information via the use of a registration contract, token contract, and token transactions, as well as to preserve data integrity by the use of PoA due to its high processing efficiency. The routing protocol of the proposed framework in [155] exploits a reinforcement learning algorithm to dynamically provide trusted routes. The results in [155] confirm that the average packet delay is reduced by 81% compared to state of the art techniques thanks to the trusted queue length information released in the proposed framework [155], while the use of PoA helps to reduce token transaction latency. A PoA-based BC was considered in the proposed framework of [156] as well, which utilized a deep learning selection model through CNN to provide the validators required for the PoA smart contract instead of randomly selecting them. The proposed PoA–DL consensus mechanism was shown to require a steady latency that is less than the average transaction delays of the state-of-the-art techniques, and enhances the transaction processing capacity due to the preselected and limited number of validators. Another deep learning method, referred to as Fully Decentralized Generative Adversarial Network (FDGAN), was proposed in [157] in conjunction with GAN, IDS, and BC to design a new routing protocol named Block Chain enabled secured Routing Protocol (GBCRP).

Malicious node detection techniques using integrated ML and BC methods were the focus in [122,152–154]. The isolated forest algorithm anomaly detection model was studied in [153]; this model it is not computationally demanding and can deliver good detection performance, especially in the case of high-volume and high-dimensional processed data. The BC helps to ensure safe storage and adequate updating of the isolated forest global detection model by providing the required trusted blocks (isolated trees) to form the model. The results reported by [153] indicate that the proposed anomaly detection model integrating BC and the isolated forest algorithm can achieve a high detection level and accuracy rate for all types of attacks while requiring less communication and storage overhead compared to other similar BC-based anomaly detection models, as it only stores the detection model and not the detection results. A joint identity management and secure

routing model was proposed in [154], in which the GA-SVM and GA-DT ML techniques were examined for the detection of malicious nodes. It was shown that GA-SVM is better than GA-DT in terms of detection accuracy; the outcome of the GA-DT process determines whether the node continues to be involved in the routing process or whether its registration in the BC network is revoked, and the safety of the routing transactions is secured using the PoA consensus mechanism. It was shown in [154] that when removing MNs, the packet delivery rate increased to 99.72%. Another consensus mechanism known as Verifiable Byzantine Fault Tolerance (VBFT) was used to validate transactions in [152], while the use of the HGB–ML classifier was proposed for detecting MN. Furthermore, [152] proposed storing data associated with normal nodes in an Interplanetary File System (IFS) to generate hashed chunks that can be then stored in the BC. Extensive comparisons were performed in [152], showing high precision of at least 98% obtained using HGB, which is more than could be achieved by its counterparts, and further demonstrating the lower transaction costs of VBFT compared to PoW. Our prior work in [122] proposed a BC-based identity management and secure authentication mechanism using a Gaussian NB detection module to mitigate possible internal DoS attacks targeting CH nodes.

10.2. Research Challenges

Developing a lightweight integrated framework that combines BC and ML while being WSN-compatible is a research area that remains in its infancy, and many open issues and challenges must be carefully addressed. The challenges associated with such systems combine the challenges related to each individual technology. Key technical challenges can be segmented into integration performance, scalability, lightweight architectures and schemes, managing network resources, legal issues, and vulnerabilities.

- Integration performance: BC and ML integration performance depends on each technology's performance; however, having both technologies operating within the same system rsises the idea of using each technology to improve the functional performance of the other. For example, ML model detection performance can be degraded by data tampering. In this regard, BC can protect the data transactions used to train the ML models along with the recorded decisions (i.e., output) of attack classification with confidence, disallowing tampering. These records can be reviewed and audited at any time by authorized nodes, and can be used to improve future ML detection decisions. In this way, incremental ML models can improve their future decision-making to detect novel attacks and handle drift in networks that change dynamically over time [65].
- Scalability: a measure of how well systems are used in conjunction with WSNs, scalability is related to network capacity in terms of the number of nodes that can join and the transaction volume that can be generated and processed over the network. The selection of the BC type and consensus mechanism highly affect scalability. For instance, the PBFT and PoA consensus mechanisms can improve transaction throughput compared to PoW, which usually supports only a few dozen transactions per second. Frequent authentication and peer trust requirements coupled with increased ledger size as the number of nodes and data increase present a challenge when aiming for a scalable ML–BC integrated security framework; however, many solutions have been presented in the literature that support scalability when employing BC technology. Among these solutions is the use of a hybrid BC, which utilizes a public BC connected to multiple private BCs wherein each private BC operates with one WSN. This structure limits the transaction volume and size of the ledger, ensuring better scalability. Among the known consensus mechanisms, voting or multiparty consensus works better with private BCs, and their combination is a candidate for use in cooperative WSNs. Another consensus mechanism is Proof-of-Authentication (PoAh), proposed in [158] for resource-scarce networks, adn which could be tested for WSNs. ML algorithms, on the other hand, can be used to code smart contracts for a more scalable approach to effective detection of malicious nodes.

- Lightweight schemes: to reduce overhead, the development and refinement of lightweight BC–ML integrated schemes while maintaining the same desired security level is essential. Deploying BC involves many elements, such as trust, authentication, access control, smart contracts, and consensus mechanisms, and each element can be implemented using a variety of options. The complexity of a BC can be refined by considering lightweight schemes in terms of storage, processing, and communication for each element involved in the deployed BC. For instance, in [152] the authors suggested Interplanetary File System (IPFS) to record the detection process, with the aim of reducing the cost of data storage in WSN; however, they did not consider the communication overhead required to upload and download data between IPFS and BS. In terms of consensus, PBFT and PoA are preferable, as they offer reduced computation and delay compared to PoW.
- Vulnerability: the ultimate goal of combining ML and BC into one system is the potential increase in security level; however, this integration does not completely eliminate threats. The root of these possible threats can be understood by considering that even though data my be safely protected by BC, it could be susceptible to tampering before it is securely recorded in the ledger.
  Considering the two approaches for BC implementation, namely, public and private, a public BC is open and accessible to all nodes, whereas a private BC is not. Therefore, a is preferable when higher levels of security are desired [159]. However, private BCs limit access to the large amount of data required to develop an efficient ML model, especially with the amount of continuously developed attack types, which makes an ML–BC integrated system vulnerable to newly developed attacks. Other possible threats might be due to malfunctioning or faulty sensors, or even sensors equipped with extra hardware allowing them to be operated maliciously, and which cannot be detected unless physically tested. These challenges add up when considering that nodes can become malicious and threaten the network security after joining the network. In addition, smart contracts can be vulnerable to possible smart contract-based attacks due to bugs in the smart contract code. ML can be used for smart contract verification and vulnerability detection [160].
- Managing network resources: limited-resource sensor nodes represent a key technical challenge when developing an ML–BC integrated solution considering encryption, trust and authentication, and validation of transactions through consensus. The ledger grows exponentially over time, and eventually may not fit within a node's memory. These technical challenges in terms of storage and processing translate into high power consumption, extending across all aspects of system design. The authors of [161] suggested a solution to this problem by switching to symmetric instead of asymmetric BC encryption in order to simplify the system's computational complexity. The computational complexity can be reduced using a simplified method for hash function calculation, such as SHA-256 [161]. Another proposed direction is dedicating specific nodes with high capabilities, such as CHs and BS nodes, for ledger storage, with other nodes only keeping the constant-length hash value of the data in the ledger to be referenced when needed. In addition, old data can be migrated from the CHs and BS toward the IoT cloud or external storage (i.e., IPFS).
- Legal issues: proliferation of different standards or a lack of security regulation can represent a challenge when designing systems involving two different technologies. Setting standards for such integrated solutions can potentially be done at the level of manufacturing and fabrication, that is, at the sensor stage.

Overall, a substantial amount of future research needs to be directed toward designing a robust ML–BC integrated solution to secure WSNs before they can be expected to work smoothly. A lightweight framework must be designed that considers sensor resource constraints and is able to effectively secure WSNs in terms of establishing trust in a trustless environment. Specially-developed consensus mechanisms, application-specific smart contracts, simple transaction verification, an alternative to block mining, and optimized

architectures that balance computation and communication consumption between nodes are vital to promoting such integration in WSN applications. In this regard, we propose the BC–ML integrated system depicted in Figure 15. The proposed system includes an ML detection model that detects the malicious behaviour of nodes using neighboring information. The ML model can identify unknown types of attacks by recognizing any deviation from the normal operation of a system as malicious [162], which allows it to use transfer learning to detect new and unknown attacks by transferring its knowledge of known attacks [163]. Concurrently, a BC-based prevention model avoids possible attempts by malicious nodes to modify their data. The BC records the ML detection process securely on the BC ledger in order to maintain its integrity. Furthermore, an smart contract is used for identity management to prevent malicious nodes from becoming authorized to access the BC network (if they are newly deployed) or to revoked their access (should malicious behaviour be detected). In addition, a trust smart contract ensures end-to-end trustworthiness between communicating nodes and limits the negative impact caused by attacks to only the affected part of the network, specifically when a cluster-based architecture is employed [134]. Smart contracts can host ML models to establish trust between nodes, making smart contracts more effective. It has been proposed to use ML models to detect smart contract-based attacks or vulnerable smart contracts deployed by malicious nodes; however, studies have revealed that smart contracts may not be able to process ML tasks with high computational needs [164]. The proposed overall BC structure is a multi-layer or hybrid one, with private BCs deployed for internal authentication in the network and a public BC deployed between the BS and IoT cloud.

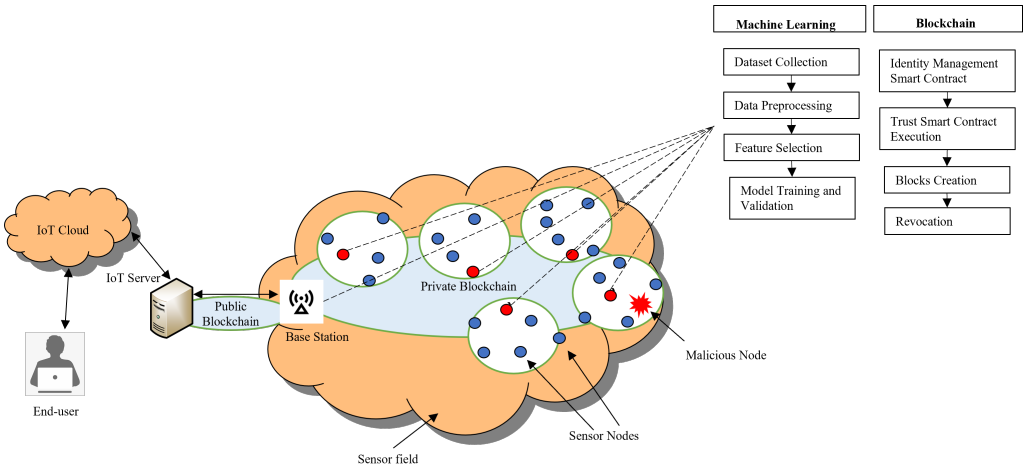

Figure 15. Proposed integrated BC–ML system for use in WSNs.

11. Conclusions

Several countermeasures to secure WSNs have been considered in this review, and extensive research efforts have been made to address the related security threats. However, at present these networks cannot manage the computational overhead necessary to implement many of the proposed defensive strategies. ML and BC are two promising technologies that we have focused on in this study for ensuring secure WSNs. In this paper, we have aimed to investigate the integration of both technologies towards a lightweight security framework for WSNs. Our review began by discussing existing surveys on ML and BC in WSN contexts, then provided a taxonomy of ML and BC approaches for WSN-related cyberattack detection and prevention. We next discussed related work and open issues for future research associated with both technologies. Finally, we illustrated the integration of ML and BC to secure WSNs, surveyed related work, and discussed the associated challenges. Finally, we ended our review by proposing the use of an integrated ML and BC system in two lines of defense to enhance the security of WSNs. In our future work, we will

consider the implementation of the proposed framework and examine the performance of the integrated system with the goal of enhancing the security of WSNs.

Author Contributions: Conceptualization, S.I. and D.W.D.; Methodology, S.I. and D.W.D.; visualization, S.I. and D.W.D.; Supervision, H.R.; Writing—original draft, S.I. and D.W.D.; Writing—review, S.I., D.W.D. and H.R.; Writing—editing, H.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abbas, G.; Mehmood, A.; Carsten, M.; Epiphaniou, G.; Lloret, J. Safety, Security and Privacy in Machine Learning Based Internet of Things. *J. Sens. Actuator Netw.* **2022**, *11*, 38.
2. Bajaj, K.; Sharma, B.; Singh, R. Integration of WSN with IoT applications: A vision, architecture, and future challenges. In *Integration of WSN and IoT for Smart Cities*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 79–102.
3. Pavithran, D.; Shaalan, K.; Al-Karaki, J.N.; Gawanmeh, A. Towards building a blockchain framework for IoT. *Clust. Comput.* **2020**, *23*, 2089–2103.
4. Sinha, P.; Jha, V.K.; Rai, A.K.; Bhushan, B. Security vulnerabilities, attacks and countermeasures in wireless sensor networks at various layers of OSI reference model: A survey. In Proceedings of the 2017 International Conference on Signal Processing and Communication (ICSPC), Coimbatore, India, 28–29 July 2017; pp. 288–293.
5. Panda, M. Security in wireless sensor networks using cryptographic techniques. *Am. J. Eng. Res. (AJER)* **2014**, *3*, 50–56.
6. Hussain, F.; Hussain, R.; Hassan, S.A.; Hossain, E. Machine learning in IoT security: Current solutions and future challenges. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1686–1721.
7. Xu, L.D.; Lu, Y.; Li, L. Embedding Blockchain Technology Into IoT for Security: A Survey. *IEEE Internet Things J.* **2021**, *8*, 10452–10473. https://doi.org/10.1109/JIOT.2021.3060508.
8. Kumar, D.P.; Amgoth, T.; Annavarapu, C.S.R. Machine learning algorithms for wireless sensor networks: A survey. *Inf. Fusion* **2019**, *49*, 1–25.
9. Alsheikh, M.A.; Lin, S.; Niyato, D.; Tan, H.P. Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1996–2018. https://doi.org/10.1109/COMST.2014.2320099.
10. Bout, E.; Loscri, V.; Gallais, A. How Machine Learning Changes the Nature of Cyberattacks on IoT Networks: A Survey. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 248–279. https://doi.org/10.1109/COMST.2021.3127267.
11. Tahsien, S.M.; Karimipour, H.; Spachos, P. Machine learning based solutions for security of Internet of Things (IoT): A survey. *J. Netw. Comput. Appl.* **2020**, *161*. https://doi.org/10.1016/j.jnca.2020.102630.
12. da Costa, K.A.P.; Papa, J.P.; Lisboa, C.O.; Munoz, R.; de Albuquerque, V.H.C. Internet of Things: A survey on machine learning-based intrusion detection approaches. *Comput. Netw.* **2019**, *151*, 147–157.
13. Ahmad, R.; Alsmadi, I. Machine learning approaches to IoT security: A systematic literature review. *Internet Things* **2021**, *14*, 100365. https://doi.org/10.1016/j.iot.2021.100365.
14. Haji, S.H.; Ameen, S.Y. Attack and Anomaly Detection in IoT Networks using Machine Learning Techniques: A Review. *Asian J. Res. Comput. Sci.* **2021**, *9*, 30–46. https://doi.org/10.9734/ajrcos/2021/v9i230218.
15. Faraj, O.; Megias, D.; Ahmad, A.M.; Garcia-Alfaro, J. Taxonomy and challenges in machine learning-based approaches to detect attacks in the internet of things. In Proceedings of the 15th International Conference on Availability, Reliability and Security, Virtual, 25–28 August 2020; pp. 1–10.
16. Mamdouh, M.; Elrukhsi, M.A.I.; Khattab, A. Securing the internet of things and wireless sensor networks via machine learning: A survey. In Proceedings of the 2018 International Conference on Computer and Applications (ICCA), Beirut, Lebanon, 25–26 August 2018; pp. 215–218.
17. Mehta, A.; Sandhu, J.K.; Sapra, L. Machine Learning in Wireless Sensor Networks: A Retrospective. In Proceedings of the 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan, India, 6–8 November 2020; pp. 328–331. https://doi.org/10.1109/PDGC50313.2020.9315767.
18. Baraneetharan, E. Role of machine learning algorithms intrusion detection in WSNs: A survey. *J. Inf. Technol.* **2020**, *2*, 161–173.
19. Gunduz, S.; Arslan, B.; Demirci, M. A Review of Machine Learning Solutions to Denial-of- Services Attacks in Wireless Sensor Networks. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015. https://doi.org/10.1109/ICMLA.2015.202.
20. Kim, T.; Vecchietti, L.F.; Choi, K.; Lee, S.; Har, D. Machine Learning for Advanced Wireless Sensor Networks: A Review. *IEEE Sens. J.* **2021**, *21*, 12379–12397. https://doi.org/10.1109/JSEN.2020.3035846.
21. Ramotsoela, D.; Abu-Mahfouz, A.; Hancke, G. A survey of anomaly detection in industrial wireless sensor networks with critical water system infrastructure as a case study. *Sensors* **2018**, *18*, 2491.

22. Ahmad, R.; Wazirali, R.; Abu-Ain, T. Machine Learning for Wireless Sensor Networks Security: An Overview of Challenges and Issues. *Sensors* **2022**, *22*, 4730.
23. Jesus, E.F.; Chicarino, V.R.; De Albuquerque, C.V.; Rocha, A.A.A. A Survey of How to Use Blockchain to Secure Internet of Things and the Stalker Attack. *Secur. Commun. Netw.* **2018**, *2018*, 9675050. https://doi.org/10.1155/2018/9675050.
24. Liao, Z.; Pang, X.; Zhang, J.; Xiong, B.; Wang, J. Blockchain on Security and Forensics Management in Edge Computing for IoT: A Comprehensive Survey. *IEEE Trans. Netw. Serv. Manag.* **2021**, *19*, 1159–1175. https://doi.org/10.1109/TNSM.2021.3122147.
25. Sengupta, J.; Ruj, S.; Das Bit, S. A Comprehensive Survey on Attacks, Security Issues and Blockchain Solutions for IoT and IIoT. *J. Netw. Comput. Appl.* **2020**, *149*, 102481. https://doi.org/10.1016/j.jnca.2019.102481.
26. Khan, M.A.; Salah, K. IoT security: Review, blockchain solutions, and open challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411. https://doi.org/10.1016/j.future.2017.11.022.
27. Darla, S.; Naveena, C. Survey on Securing Internet of Things through Block chain Technology. In Proceedings of the 2022 International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 16–18 March 2022; pp. 836–844. https://doi.org/10.1109/ICEARS53579.2022.9752316.
28. Uddin, M.A.; Stranieri, A.; Gondal, I.; Balasubramanian, V. A survey on the adoption of blockchain in IoT: Challenges and solutions. *Blockchain Res. Appl.* **2021**, *2*, 100006. https://doi.org/10.1016/j.bcra.2021.100006.
29. Pohrmen, F.H.; Das, R.K.; Saha, G. Blockchain-based security aspects in heterogeneous Internet-of-Things networks: a survey. *Trans. Emerg. Telecommun. Technol.* **2019**, *30*, e3741.
30. Miglani, A.; Kumar, N. Blockchain management and machine learning adaptation for IoT environment in 5G and beyond networks: A systematic review. *Comput. Commun.* **2021**, *178*, 37–63. https://doi.org/10.1016/j.comcom.2021.07.009.
31. Matin, M.A.; Islam, M.M. Overview of wireless sensor network. *Wirel. Sens.-Netw.-Technol. Protoc.* **2012**, *1*, 3.
32. Khan, Z.A.; Samad, A. A study of machine learning in wireless sensor network. *Int. J. Comput. Netw. Appl.* **2017**, *4*, 105–112.
33. Rehana, J. Security of wireless sensor network. In Proceedings of the Seminar on Internetworking, Helsinki University of Technology, Glasgow, UK, 24–28 August 2009.
34. Sora, D. Security Issues in Wireless Sensor Networks. *Int. J. Online Biomed. Eng. (IJOE)* **2010**, *6*, 26–30. https://doi.org/10.3991/ijoe.v6i4.1466.
35. Patel, N.R.; Kumar, S. Wireless Sensor Networks' Challenges and Future Prospects. In Proceedings of the 2018 International Conference on System Modeling & Advancement in Research Trends (SMART), Moradabad, India, 23–24 November 2018; pp. 60–65. https://doi.org/10.1109/SYSMART.2018.8746937.
36. de Farias, C.M.; Pirmez, L.; Delicato, F.C.; Pires, P.F.; Guerrieri, A.; Fortino, G.; Cauteruccio, F.; Terracina, G. A multisensor data fusion algorithm using the hidden correlations in Multiapplication Wireless Sensor data streams. In Proceedings of the 2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC), Calabria, Italy, 16–18 May 2017; pp. 96–102. https://doi.org/10.1109/ICNSC.2017.8000074.
37. Karray, F.; Jmal, M.W.; Garcia-Ortiz, A.; Abid, M.; Obeid, A.M. A comprehensive survey on wireless sensor node hardware platforms. *Comput. Netw.* **2018**, *144*, 89–110. https://doi.org/10.1016/j.comnet.2018.05.010.
38. Xie, H.; Yan, Z.; Member, S.; Yao, Z. Data Collection for Security Measurement in Wireless Sensor Networks: A Survey. *IEEE Internet Things J.* **2019**. *6*, 2205–2224.
39. Alam, S.; De, D. Analysis of security threats in wireless sensor network. *arXiv* **2014**, arXiv:1406.0298.
40. Walters, J.P.; Liang, Z.; Shi, W.; Chaudhary, V. Wireless sensor network security: A survey. In *Security in distributed, Grid, and Pervasive Computing*; Auerbach Publications: Boca Raton, FL, USA, 2006; pp. 208–222.
41. Chapter 16—Wireless Sensor Network Security. In *Computer and Information Security Handbook*, 2nd ed.; Vacca, J.R., Ed.; Morgan Kaufmann: Boston, MA, USA, 2013; pp. 301–322. https://doi.org/10.1016/B978-0-12-394397-2.00016-7.
42. Elhoseny, M.; Hassanien, A.E. Secure data transmission in WSN: An overview. In *Dynamic Wireless Sensor Networks*; Springer: Cham, Switzerland, 2019; pp. 115–143.
43. Shahzad, F.; Pasha, M.; Ahmad, A. A survey of active attacks on wireless sensor networks and their countermeasures. *arXiv* **2017**, arXiv:1702.07136.
44. Mathew, A.; Terence, J.S. A survey on various detection techniques of sinkhole attacks in WSN. In Proceedings of the 2017 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 6–8 April 2017; pp. 1115–1119.
45. Dewal, P.; Narula, G.S.; Jain, V.; Baliyan, A. Security attacks in Wireless sensor networks: A survey. In *Cyber Security*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 47–58.
46. Kaur, R.; Kaur Sandhu, J. A Study on Security Attacks in Wireless Sensor Network. In Proceedings of the 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 4–5 March 2021; pp. 850–855. https://doi.org/10.1109/ICACITE51222.2021.9404619.
47. Ismail, S.; Khoei, T.T.; Marsh, R.; Kaabouch, N. A Comparative Study of Machine Learning Models for Cyber-attacks Detection in Wireless Sensor Networks. In Proceedings of the 2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 1–4 December 2021; pp. 1–5.
48. Pruthi, V.; Mittal, K.; Sharma, N.; Kaushik, I. Network layers threats & its countermeasures in WSNs. In Proceedings of the 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 18–19 October 2019; pp. 156–163.

49. Yang, G.; Dai, L.; Wei, Z. Challenges, threats, security issues and new trends of underwater wireless sensor networks. *Sensors* **2018**, *18*, 3907. https://doi.org/10.3390/s18113907.
50. de Lima Pinto, E.M.; Lachowski, R.; Pellenz, M.E.; Penna, M.C.; Souza, R.D. A machine learning approach for detecting spoofing attacks in wireless sensor networks. In Proceedings of the 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA), Krakow, Poland, 16–18 May 2018; pp. 752–758.
51. Bhattasali, T.; Chaki, R. A Survey of Recent Intrusion Detection Systems for wireless sensor network. In *Advances in Network Security and Applications, Proceedings of the 4th International Conference, CNSA 2011, Chennai, India, 15–17 July 2011*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 268–269.
52. Tiberti, W.; Carmenini, A.; Pomante, L.; Cassioli, D. A Lightweight Blockchain-based Technique for Anti-Tampering in Wireless Sensor Networks. In Proceedings of the 2020 23rd Euromicro Conference on Digital System Design (DSD), Kranj, Slovenia, 26–28 August 2020; pp. 577–582.
53. Periyanayagi, S.; Sumathy, V. Swarm-based defense technique for tampering and cheating attack in WSN using CPHS. *Pers. Ubiquitous Comput.* **2018**, *22*, 1165–1179.
54. Numan, M.; Subhan, F.; Khan, W.Z.; Hakak, S.; Haider, S.; Reddy, G.T.; Jolfaei, A.; Alazab, M. A Systematic Review on Clone Node Detection in Static Wireless Sensor Networks. *IEEE Access* **2020**, *8*, 65450–65461. https://doi.org/10.1109/ACCESS.2020.2983091.
55. Gupta, S.; Verma, H.K.; Sangal, A.L. Security attacks & prerequisite for wireless sensor networks. *Int. J. Eng. Adv. Technol. (IJEAT)* **2013**, *2*, 558–566.
56. Premkumar, M.; Sundararajan, T.V. DLDM: Deep learning-based defense mechanism for denial of service attacks in wireless sensor networks. *Microprocess. Microsyst.* **2020**, *79*, 103278. https://doi.org/10.1016/j.micpro.2020.103278.
57. Mohapatra, H. Handling of Man-In-The-Middle Attack in WSN Through Intrusion Detection System. *Int. J. Emerg. Trends Eng. Res.* **2020**, *8*, 1503–1510. https://doi.org/10.30534/ijeter/2020/05852020.
58. Yahyaoui, A.; Abdellatif, T.; Attia, R. Hierarchical anomaly based intrusion detection and localization in IoT. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; pp. 108–113.
59. Somaya, H.; Tomader, M. Build a malware detection software for IOT network Using Machine learning. In Proceedings of the 4th International Conference on Networking, Information Systems & Security, Kenitra, Morocco, 1–2 April 2021; pp. 1–8.
60. Dener, M.; Al, S.; Orman, A. STLGBM-DDS: An Efficient Data Balanced DoS Detection System for Wireless Sensor Networks on Big Data Environment. *IEEE Access* **2022**, *10*, 92931–92945. https://doi.org/10.1109/ACCESS.2022.3202807.
61. Park, T.; Cho, D.; Kim, H. An effective classification for DoS attacks in wireless sensor networks. In Proceedings of the 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), Prague, Czech Republic, 3–6 July 2018; pp. 689–692.
62. Quincozes, S.E.; Kazienko, J.F. Machine learning methods assessment for denial of service detection in wireless sensor networks. In Proceedings of the 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, 2–16 June 2020; pp. 1–6.
63. Alsubaie, F.; Al-Akhras, M.; Alzahrani, H.A. Using machine learning for intrusion detection system in wireless body area network. In Proceedings of the 2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH), Riyadh, Saudi Arabia, 3–5 November 2020; pp. 100–104.
64. Alsulaiman, L.; Al-Ahmadi, S. Performance evaluation of machine learning techniques for DOS detection in wireless sensor network. *arXiv* **2021**, arXiv:2104.01963.
65. Ifzarne, S.; Tabbaa, H.; Hafidi, I.; Lamghari, N. Anomaly detection using machine learning techniques in wireless sensor networks. *J. Phys. Conf. Ser.* **2021**, *1743*, 012021.
66. Batiha, T.; Krömer, P. Design and analysis of efficient neural intrusion detection for wireless sensor networks. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6152.
67. Al-Akhras, M.; Al-Issa, A.I.; Alsahli, M.S.; Alawairdhi, M. POSTER: Feature Selection to Optimize DoS Detection in Wireless Sensor Networks. In Proceedings of the 2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH), Riyadh, Saudi Arabia, 3–5 November 2020; pp. 263–265.
68. Batiha, T.; Prauzek, M.; Krömer, P. *Intrusion Detection in Wireless Sensor Networks by an Ensemble of Artificial Neural Networks*; Springer: Singapore, 2019; Volume 142, pp. 323–333. https://doi.org/10.1007/978-981-13-8311-3_28.
69. Ismail, S.; Dawoud, D.; Reza, H. A Lightweight Multilayer Machine Learning Detection System for Cyber-attacks in WSN. In Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), Virtual, 26–29 January 2022; pp. 481–486. https://doi.org/10.1109/CCWC54503.2022.9720891.
70. Ismail, S.; Reza, H. Evaluation of Naïve Bayesian Algorithms for Cyber-Attacks Detection in Wireless Sensor Networks. In Proceedings of the 2022 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 6–9 June 2022; pp. 283–289.
71. Meng, D.; Dai, H.; Sun, Q.; Xu, Y.; Shi, T. Novel Wireless Sensor Network Intrusion Detection Method Based on LightGBM Model. *IAENG Int. J. Appl. Math.* **2022**, *52*, 1–7.
72. Luo, T.; Nagarajan, S.G. Distributed anomaly detection using autoencoder neural networks in WSN for IoT. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
73. Sherubha, P.; Amudhavalli, P.; Sasirekha, S. Clone attack detection using random forest and multi objective cuckoo search classification. In Proceedings of the 2019 International Conference on Communication and Signal Processing (ICCSP), Melmaruvathur, India, 4–6 April 2019; pp. 0450–0454.

74. Otoum, S.; Kantarci, B.; Mouftah, H. Empowering reinforcement learning on big sensed data for intrusion detection. In Proceedings of the ICC 2019-2019 IEEE international conference on communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7.
75. Subasini, C.; Karuppiah, S.; Sheeba, A.; Padmakala, S. Developing an attack detection framework for wireless sensor network-based healthcare applications using hybrid convolutional neural network. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4336.
76. Salmi, S.; Oughdir, L. CNN-LSTM Based Approach for Dos Attacks Detection in Wireless Sensor Networks. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*. https://doi.org/10.14569/IJACSA.2022.0130497.
77. Salmi, S.; Oughdir, L. Performance evaluation of deep learning techniques for DoS attacks detection in wireless sensor network. *J. Big Data* **2023**, *10*, 1–25.
78. Otoum, S.; Kantarci, B.; Mouftah, H.T. On the feasibility of deep learning in sensor network intrusion detection. *IEEE Netw. Lett.* **2019**, *1*, 68–71.
79. Hussain, K.; Xia, Y.; Onaizah, A.N.; Manzoor, T.; Jalil, K. Hybrid of WOA-ABC and Proposed CNN for Intrusion Detection System in wireless sensor networks. *Optik* **2022**, 170145. https://doi.org/10.1016/j.ijleo.2022.170145.
80. Nguyen, T.T.; Reddi, V.J. Deep reinforcement learning for cyber security. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**. https://doi.org/10.1109/TNNLS.2021.3121870.
81. Benaddi, H.; Ibrahimi, K.; Benslimane, A.; Qadir, J. A deep reinforcement learning based intrusion detection system (drl-ids) for securing wireless sensor networks and internet of things. In Proceedings of the International Wireless Internet Conference, Taichung, Taiwan, 26–27 November 2019; pp. 73–87.
82. Niknam, S.; Dhillon, H.S.; Reed, J.H. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Commun. Mag.* **2020**, *58*, 46–51.
83. Kamel, R.M.; El Mougy, A. Retrospective sensing based on federated learning in the IoT. In Proceedings of the 2020 IEEE 45th LCN Symposium on Emerging Topics in Networking (LCN Symposium), Sydney, Australia, 16–19 November 2020; pp. 150–161.
84. Kim, S.; Cai, H.; Hua, C.; Gu, P.; Xu, W.; Park, J. Collaborative anomaly detection for internet of things based on federated learning. In Proceedings of the 2020 IEEE/CIC International Conference on Communications in China (ICCC), Chongqing, China, 9–11 August 2020; pp. 623–628.
85. Mertens, J.; Galluccio, L.; Morabito, G. Federated learning through model gossiping in wireless sensor networks. In Proceedings of the 2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Bucharest, Romania, 24–28 May 2021; pp. 1–6.
86. Banerjee, J.; Maiti, S.; Chakraborty, S.; Dutta, S.; Chakraborty, A.; Banerjee, J.S. Impact of Machine Learning in Various Network Security Applications. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019; pp. 276–281. https://doi.org/10.1109/ICCMC.2019.8819811.
87. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221.
88. Zahariadis, T.; Trakadas, P.; Maniatis, S.; Karkazis, P.; Leligou, H.C.; Voliotis, S. Efficient detection of routing attacks in wireless sensor networks. In Proceedings of the 2009 16th International Conference on Systems, Signals and Image Processing, Chalkida, Greece, 18–20 June 2009; pp. 1–4.
89. Loo, C.E.; Ng, M.Y.; Leckie, C.; Palaniswami, M. Intrusion detection for routing attacks in sensor networks. *Int. J. Distrib. Sens. Netw.* **2006**, *2*, 313–332.
90. Amouri, A.; Alaparthy, V.T.; Morgera, S.D. Cross layer-based intrusion detection based on network behavior for IoT. In Proceedings of the 2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON), Sand Key, FL, USA, 9–10 April 2018; pp. 1–4. https://doi.org/10.1109/WAMICON.2018.8363921.
91. Pande, S.; Khamparia, A.; Gupta, D. Feature selection and comparison of classification algorithms for wireless sensor networks. *J. Ambient. Intell. Humaniz. Comput.* **2021**, 1–13. https://doi.org/10.1007/s12652-021-03411-6.
92. Almomani, I.; Al-Kasasbeh, B.; Al-Akhras, M. WSN-DS: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks. *J. Sens.* **2016**, *2016*. https://doi.org/10.1155/2016/4731953.
93. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
94. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* **2018**, *1*, 108–116.
95. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6.
96. Tharwat, A. Classification assessment methods. *Appl. Comput. Inform.* **2018**, *17*, 168–192. https://doi.org/10.1016/j.aci.2018.08.003.
97. Panda, M.; Abd Allah, A.M.; Hassanien, A.E. Developing an Efficient Feature Engineering and Machine Learning Model for Detecting IoT-Botnet Cyber Attacks. *IEEE Access* **2021**, *9*, 91038–91052.

98. Alsahli, M.S.; Almasri, M.M.; Al-Akhras, M.; Al-Issa, A.I.; Alawairdhi, M. Evaluation of Machine Learning Algorithms for Intrusion Detection System in WSN. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 617–626. https://doi.org/10.14569/IJACSA.2021.0120574.
99. Haber, S.; Stornetta, W.S. How to Time-Stamp a Digital Document. In *Proceedings of the Advances in Cryptology-CRYPTO' 90; Menezes*; Menezes, A.J., Vanstone, S.A., Eds.; Springer: Berlin/Heidelberg, Germany, 1991; pp. 437–455.
100. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. *Decentralized Bus. Rev.* **2008**, 21260. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 21 August 2018).
101. Wang, X.; Zha, X.; Ni, W.; Liu, R.P.; Guo, Y.J.; Niu, X.; Zheng, K. Survey on blockchain for Internet of Things. *Comput. Commun.* **2019**, *136*, 10–29. https://doi.org/10.1016/J.COMCOM.2019.01.006.
102. Gao, W.; Hatcher, W.G.; Yu, W. A Survey of Blockchain: Techniques, Applications, and Challenges. In Proceedings of the 2018 27th International Conference on Computer Communication and Networks (ICCCN), Hangzhou, China, 30 July–2 August 2018; pp. 1–11. https://doi.org/10.1109/ICCCN.2018.8487348.
103. Zhang, S.; Lee, J.H. Analysis of the main consensus protocols of blockchain. *ICT Express* **2020**, *6*, 93–97. https://doi.org/10.1016/J.ICTE.2019.08.001.
104. Gagneja, K.; Gagneja, K.; Kiefer, R. Security Protocol for Internet of Things (IoT): Blockchain-based Implementation and Analysis. In Proceedings of the 2020 Sixth International Conference on Mobile And Secure Services (MobiSecServ), Miami, FL, USA, 22–23 February 2020. https://doi.org/10.1109/MobiSecServ48690.2020.9042948.
105. Monrat, A.A.; Schelén, O.; Andersson, K. A survey of blockchain from the perspectives of applications, challenges, and opportunities. *IEEE Access* **2019**, *7*, 117134–117151.
106. Hsiao, S.J. Employing Blockchain Technology to Strengthen Security of Wireless Sensor Networks. *IEEE Access* **2021**, *9*, 72326–72341. https://doi.org/10.1109/ACCESS.2021.3079708.
107. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. Blendcac: A blockchain-enabled decentralized capability-based access control for iots. In Proceedings of the IEEE 2018 International Congress on Cybermatics: 2018 IEEE Conferences on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, iThings/Gree Halifax, Canada, 30 July–3 August 2018; pp. 1027–1034, [1804.09267]. https://doi.org/10.1109/Cybermatics_2018.2018.00191.
108. Khalil, A.A.; Franco, J.; Parvez, I.; Uluagac, S.; Rahman, M.A. A Literature Review on Blockchain-enabled Security and Operation of Cyber-Physical Systems. In Proceedings of the 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC), Los Alamitos, CA, USA, 27 June–1 July 2022. [2107.07916].
109. Cui, Z.; Xue, F.; Zhang, S.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. A Hybrid BlockChain-Based Identity Authentication Scheme for Multi-WSN. *IEEE Trans. Serv. Comput.* **2020**, *13*, 241–251. https://doi.org/10.1109/TSC.2020.2964537.
110. Mamdouh, M.; Awad, A.I.; Khalaf, A.A.; Hamed, H.F. Authentication and Identity Management of IoHT Devices: Achievements, Challenges, and Future Directions. *Comput. Secur.* **2021**, *111*, 102491. https://doi.org/10.1016/j.cose.2021.102491.
111. Salimitari, M.; Chatterjee, M. A Survey on Consensus Protocols in Blockchain for IoT Networks. *arXiv* **2018**, 1–15, arXiv:1809.05613v4. [1809.05613].
112. Mohanta, B.K.; Panda, S.S.; Jena, D. An overview of smart contract and use cases in blockchain technology. In Proceedings of the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 10–12 July 2018; pp. 1–4.
113. Teslya, N.; Ryabchikov, I. Blockchain platforms overview for industrial IoT purposes. In Proceedings of the Conference of Open Innovation Association, FRUCT, Jyvaskyla, Finland, 15–18 May 2018; pp. 250–256. https://doi.org/10.23919/FRUCT.2018.8468276.
114. Ismail, S.; Reza, H.; Zadeh, H.K.; Vasefi, F. A Blockchain-based IoT Security Solution Using Multichain. In Proceedings of the 2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 8–11 March 2023; pp. 1105–1111. https://doi.org/10.1109/CCWC57344.2023.10099128.
115. Alkurdi, F.; Elgendi, I.; Munasinghe, K.S.; Sharma, D.; Jamalipour, A. Blockchain in IoT Security: A Survey. In Proceedings of the 2018 28th International Telecommunication Networks and Applications Conference, ITNAC 2018, Sydney, Australia, 21–23 November 2018; pp. 1–4. https://doi.org/10.1109/ATNAC.2018.8615409.
116. Liu, Y.; Yu, F.R.; Li, X.; Ji, H.; Leung, V.C. Blockchain and Machine Learning for Communications and Networking Systems. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1392–1431. https://doi.org/10.1109/COMST.2020.2975911.
117. Honar Pajooh, H.; Rashid, M.; Alam, F.; Demidenko, S. Hyperledger fabric blockchain for securing the edge internet of things. *Sensors* **2021**, *21*, 359.
118. Tian, Y.; Wang, Z.; Xiong, J.; Ma, J. A Blockchain-Based Secure Key Management Scheme With Trustworthiness in DWSNs. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6193–6202. https://doi.org/10.1109/TII.2020.2965975.
119. Goyat, R.; Kumar, G.; Alazab, M.; Saha, R.; Thomas, R.; Rai, M.K. A secure localization scheme based on trust assessment for WSNs using blockchain technology. *Future Gener. Comput. Syst.* **2021**, *125*, 221–231.
120. Guerrero-Sanchez, A.E.; Rivas-Araiza, E.A.; Gonzalez-Cordoba, J.L.; Toledano-Ayala, M.; Takacs, A. Blockchain mechanism and symmetric encryption in a wireless sensor network. *Sensors* **2020**, *20*, 2798. https://doi.org/10.3390/s20102798.
121. Rathee, G.; Balasaraswathi, M.; Chandran, K.P.; Gupta, S.D.; Boopathi, C. A secure IoT sensors communication in industry 4.0 using blockchain technology. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 533–545.

122. Ismail, S.; Dawoud, D.; Reza, H. Towards A Lightweight Identity Management and Secure Authentication for IoT Using Blockchain. In Proceedings of the 2022 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 6–9 June 2022; pp. 77–83. https://doi.org/10.1109/AIIoT54504.2022.9817349.
123. Miraz, M.H. Blockchain of things (BCoT): The fusion of blockchain and IoT technologies. In *Advanced Applications of Blockchain Technology*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 141–159.
124. Biswas, S.; Sharif, K.; Li, F.; Nour, B.; Wang, Y. A scalable blockchain framework for secure transactions in IoT. *IEEE Internet Things J.* **2019**, *6*, 4650–4659. https://doi.org/10.1109/JIOT.2018.2874095.
125. Kushch, S.; Prieto-Castrillo, F. A rolling blockchain for a dynamic WSNs in a smart city. *arXiv* **2018**, *1*, 1–8. arXiv:1806.11399.
126. Lao, L.; Li, Z.; Hou, S.; Xiao, B.; Guo, S.; Yang, Y. A survey of IoT applications in blockchain systems: Architecture, consensus, and traffic modeling. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 1–32.
127. Zamani, M.; Movahedi, M.; Raykova, M. RapidChain: Scaling blockchain via full sharding. In Proceedings of the ACM Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018. https://doi.org/10.1145/3243734.3243853.
128. Cherupally, S.R.; Boga, S.; Podili, P.; Kataoka, K. Lightweight and Scalable DAG based distributed ledger for verifying IoT data integrity. *Int. Conf. Inf. Netw.* **2021**, *2021*, 267–272. https://doi.org/10.1109/ICOIN50884.2021.9334000.
129. Buldin, I.D.; Gorodnichev, M.G.; Makhrov, S.S.; Denisova, E.N. Next Generation Industrial Blockchain-Based Wireless Sensor Networks. In Proceedings of the 2018 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF), Saint Petersburg, Russia, 3–7 June 2018; pp. 1–5. https://doi.org/10.1109/WECONF.2018.8604408.
130. Baig, M.A.; Ali Sunny, D.; Alqahtani, A.; Alsubai, S.; Binbusayyis, A.; Muzammal, M. A Study on the Adoption of Blockchain for IoT Devices in Supply Chain. *Comput. Intell. Neurosci.* **2022**, *2022*, 9228982.
131. Goyal, H.; Saha, S. Reli: Real-time lightweight byzantine consensus in low-power iot-systems. In Proceedings of the 2022 18th International Conference on Network and Service Management (CNSM), Thessaloniki, Greece, 31 October–4 November 2022; pp. 275–281.
132. Gopalakrishnan, K. Security vulnerabilities and issues of traditional wireless sensors networks in IoT. In *Principles of Internet of Things (IoT) Ecosystem: Insight Paradigm*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 519–549.
133. Waheed, N.; He, X.; Ikram, M.; Usman, M.; Hashmi, S.S.; Usman, M. Security and Privacy in IoT Using Machine Learning and Blockchain: Threats and Countermeasures. *ACM Comput. Surv.* **2021**, *53*, 1–37. https://doi.org/10.1145/3417987.
134. Marchang, J.; Ibbotson, G.; Wheway, P. Will blockchain technology become a reality in sensor networks? In Proceedings of the 2019 Wireless Days (WD), Manchester, UK, 24–26 April 2019, pp. 1–4.
135. Shammar, E.A.; Zahary, A.T.; Al-Shargabi, A.A. A survey of IoT and blockchain integration: Security perspective. *IEEE Access* **2021**, *9*, 156114–156150.
136. She, W.; Liu, Q.; Tian, Z.; Chen, J.S.; Wang, B.; Liu, W. Blockchain trust model for malicious node detection in wireless sensor networks. *IEEE Access* **2019**, *7*, 38947–38956.
137. Arifeen, M.M.; Al Mamun, A.; Ahmed, T.; Kaiser, M.S.; Mahmud, M. A Blockchain-Based Scheme for Sybil Attack Detection in Underwater Wireless Sensor Networks. In *Proceedings of International Conference on Trends in Computational and Cognitive Engineering*; Kaiser, M.S., Bandyopadhyay, A., Mahmud, M., Ray, K., Eds.; Springer: Singapore, 2021; pp. 467–476.
138. Chanana, R.; Singh, A.K.; Killa, R.; Agarwal, S.; Mehra, P.S. Blockchain Based Secure Model for Sensor Data in Wireless Sensor Network. In Proceedings of the 2020 6th International Conference on Signal Processing and Communication (ICSC), Noida, India, 5–7 March 2020; pp. 288–293. https://doi.org/10.1109/ICSC48311.2020.9182776.
139. Mubarakali, A. An efficient authentication scheme using blockchain technology for wireless sensor networks. *Wirel. Pers. Commun.* **2021**, *1*, 1–15.
140. Sivaganesan, D. A data driven trust mechanism based on blockchain in IoT sensor networks for detection and mitigation of attacks. *J. Trends Comput. Sci. Smart Technol. (TCSST)* **2021**, *3*, 59–69.
141. Karakoç, E.; Çeken, C. Black hole attack prevention scheme using a blockchain-block approach in SDN-enabled WSN. *Int. J. Hoc Ubiquitous Comput.* **2021**, *37*, 37–49.
142. Soltani, R.; Saxena, L.; Joshi, R.; Sampalli, S. Protecting Routing Data in WSNs with use of IOTA Tangle. *Procedia Comput. Sci.* **2022**, *203*, 197–204.
143. Javed, S.; Khan, M.A.; Abdullah, A.M.; Alsirhani, A.; Alomari, A.; Noor, F.; Ullah, I. An Efficient Authentication Scheme Using Blockchain as a Certificate Authority for the Internet of Drones. *Drones* **2022**, *6*, 264. https://doi.org/10.3390/drones6100264.
144. Lazrag, H.; Chehri, A.; Saadane, R.; Rahmani, M.D. Efficient and secure routing protocol based on Blockchain approach for wireless sensor networks. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6144. https://doi.org/10.1002/cpe.6144.
145. Awan, S.; Javaid, N.; Ullah, S.; Khan, A.U.; Qamar, A.M.; Choi, J.G. Blockchain Based Secure Routing and Trust Management in Wireless Sensor Networks. *Sensors* **2022**, *22*, 411.
146. Chen, Y.; Yang, X.; Li, T.; Ren, Y.; Long, Y. A blockchain-empowered authentication scheme for worm detection in wireless sensor network. *Digit. Commun. Netw.* **2022** . https://doi.org/10.1016/j.dcan.2022.04.007.
147. Almaiah, M.A., A New Scheme for Detecting Malicious Attacks in Wireless Sensor Networks Based on Blockchain Technology. In *Artificial Intelligence and Blockchain for Future Cybersecurity Applications*; Maleh, Y., Baddi, Y., Alazab, M., Tawalbeh, L., Romdhani, I., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 217–234. https://doi.org/10.1007/978-3-030-74575-2_12.

148. Moinet, A.; Darties, B.; Baril, J.L. Blockchain based trust & authentication for decentralized sensor networks. *arXiv* **2017**, arXiv:1706.01730.
149. Kim, T.H.; Goyat, R.; Rai, M.K.; Kumar, G.; Buchanan, W.J.; Saha, R.; Thomas, R. A novel trust evaluation process for secure localization using a decentralized blockchain in wireless sensor networks. *IEEE Access* **2019**, *7*, 184133–184144.
150. Pundir, S.; Wazid, M.; Singh, D.P.; Das, A.K.; Rodrigues, J.J.P.C.; Park, Y. Intrusion detection protocols in wireless sensor networks integrated to Internet of Things deployment: Survey and future challenges. *IEEE Access* **2019**, *8*, 3343–3363.
151. Revanesh, M.; Sridhar, V. A trusted distributed routing scheme for wireless sensor networks using blockchain and meta-heuristics-based deep learning technique. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4259.
152. Nouman, M.; Qasim, U.; Nasir, H.; Almasoud, A.; Imran, M.; Javaid, N. Malicious Node Detection using Machine Learning and Distributed Data Storage using Blockchain in WSNs. *IEEE Access*, 2023, *11*, 6106–6121.
153. Yang, X.; Chen, Y.; Qian, X.; Li, T.; Lv, X. BCEAD: A blockchain-empowered ensemble anomaly detection for wireless sensor network via isolation forest. *Secur. Commun. Netw.* **2021**, *2021*, 9430132.
154. Sajid, M.B.E.; Ullah, S.; Javaid, N.; Ullah, I.; Qamar, A.M.; Zaman, F. Exploiting Machine Learning to Detect Malicious Nodes in Intelligent Sensor-Based Systems Using Blockchain. *Wirel. Commun. Mob. Comput.* **2022**, *9*, 24695–24707.
155. Yang, J.; He, S.; Xu, Y.; Chen, L.; Ren, J. A trusted routing scheme using blockchain and reinforcement learning for wireless sensor networks. *Sensors* **2019**, *19*, 970. https://doi.org/10.3390/s19040970.
156. Abd El-Moghith, I.A.; Darwish, S.M. Towards designing a trusted routing scheme in wireless sensor networks: A new deep blockchain approach. *IEEE Access* **2021**, *9*, 103822–103834. https://doi.org/10.1109/ACCESS.2021.3098933.
157. Rajasoundaran, S.; Kumar, S.; Selvi, M.; Ganapathy, S.; Rakesh, R.; Kannan, A. Machine learning based volatile block chain construction for secure routing in decentralized military sensor networks. *Wirel. Netw.* **2021**, *27*, 4513–4534.
158. Puthal, D.; Mohanty, S.P.; Nanda, P.; Kougianos, E.; Das, G. Proof-of-Authentication for Scalable Blockchain in Resource-Constrained Distributed Systems. In Proceedings of the 2019 IEEE International Conference on Consumer Electronics (ICCE), Berlin, Germany, 8–11 September 2019; pp. 1–5. https://doi.org/10.1109/ICCE.2019.8662009.
159. Mingxiao, D.; Xiaofeng, M.; Zhe, Z.; Xiangwei, W.; Qijun, C. A review on consensus algorithm of blockchain. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 2567–2572. https://doi.org/10.1109/SMC.2017.8123011.
160. Zhou, Q.; Zheng, K.; Zhang, K.; Hou, L.; Wang, X. Vulnerability Analysis of Smart Contract for Blockchain-Based IoT Applications: A Machine Learning Approach. *IEEE Internet Things J.* **2022**, *9*, 24695–24707.
161. Wang, S.Y.; Hsu, Y.J.; Hsiao, S.J. Integrating blockchain technology for data collection and analysis in wireless sensor networks with an innovative implementation. In Proceedings of the 2018 International Symposium on Computer, Consumer and Control (IS3C), Taichung, Taiwan, 6–8 December 2018; pp. 149–152.
162. Omar, S.; Ngadi, A.; Jebur, H.H. Machine learning techniques for anomaly detection: An overview. *Int. J. Comput. Appl.* **2013**, *79*, 33–41.
163. Zhao, J.; Shetty, S.; Pan, J.W.; Kamhoua, C.; Kwiat, K. Transfer learning for detecting unknown network attacks. *Eurasip J. Inf. Secur.* **2019**, *2019*, 1.
164. Lu, Y.; Tang, Q.; Wang, G. On enabling machine learning tasks atop public blockchains: A crowdsourcing approach. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 17–20 November 2018; pp. 81–88.