telecom

MDPI

Article

The Modeling and Detection of Attacks in Role-Based Self-Organized Decentralized Wireless Sensor Networks

Aleksey Meleshko and Vasily Desnitsky *

St. Petersburg Federal Research Center of the Russian Academy of Sciences, 199178 St. Petersburg, Russia; meleshko.a@iias.spb.su
* Correspondence: desnitsky@comsec.spb.ru

Abstract: This article discusses the modeling and detection of attacks in self-organizing decentralized wireless sensor networks (WSNs) that can be applied to various critical scenarios in practice. Security issues in this type of network have previously been studied to a rather poor extent. In particular, existing attack detection approaches and algorithms do not rely on the properties of self-organization and decentralization, which an attacker is able to exploit to compromise the network and its services. We propose, first, a model of a self-organizing decentralized wireless sensor network; second, a model of the attacks on such networks; third, algorithms for data collection and attack detection; and, finally, a technique for their application. The WSN model represents a formal specification of this type of network, defining the conditions and limitations of network self-organization and decentralization. The model is characterized by a proposed underlying role-based operation of network nodes and a set of their functional states. The proposed attack model covers the possible types of attacks that are relevant to a given type of WSN and are based on the exploitation of the self-organization and decentralization of the network. The developed algorithm for collecting data for attack detection presents specific types of data and their sources. The developed combined attack detection algorithm is formed of actions that detect relevant attacks on self-organizing decentralized WSNs using machine learning methods. The distinctive element of this algorithm is a set of highly specific features that are obtained by analyzing the data collected in the WSN and used to detect attacks. The proposed technique combines the constructed models and algorithms for the sake of tuning and deploying the attack detection module and the effective detection of attacks in practice. This technique specifies the main steps for the joint use of the models and algorithms and the assignment of data collection and detection parameters. The results of the experiments confirm the correctness of the constructed models, algorithms and technique due to the high values of the attack detection quality indicators. Therefore, the practical application of the proposed apparatus will facilitate improvements in the security of self-organizing decentralized WSNs. Experimental research has confirmed the practical applicability of our proposed solutions. In particular, it has shown that the proposed algorithms and the detection technique can detect both attacks implemented through the exploitation of the network's properties of decentralization/self-organization and common variations in these attacks (i.e., without exploiting the decentralization property). In general, the experimental results expose a high quality of detection, with an f1-score equal to 0.99.

Keywords: wireless sensor network; attack detection; data collection; feature construction; role-based model

Citation: Meleshko, A.; Desnitsky, V. The Modeling and Detection of Attacks in Role-Based Self-Organized Decentralized Wireless Sensor Networks. *Telecom* **2024**, *5*, 145–175. https://doi.org/10.3390/telecom5010008

Received: 19 October 2023
Revised: 16 January 2024
Accepted: 30 January 2024
Published: 9 February 2024

1. Introduction

Currently, there is an increasing spread of wireless sensor networks (WSNs) and their application in diverse areas of life, including various industrial and engineering spheres. Typically, WSNs can be used to monitor the physical environment of an enterprise or to control any specific production or other technical process. Usually, WSNs consist of a

number of small devices, i.e., sensors, which are interconnected by a wireless communication channel and are capable of collecting, processing and transmitting data in the investigated area.

Due to its specific character, the class of networks with self-organization and decentralization properties deserves special consideration. Self-organization allows the network to expand its set of nodes during operation, exclude any nodes, optimize information transmission routes and change the actual network topology of the WSN dynamically. Decentralization means that the key functions of the network are not concentrated in one place but can be distributed among the individual network nodes. Potentially, each network node can be autonomous and capable of independently performing its functions. In particular, the functions of managing the operation of the network and processing and aggregating data could be performed in parallel and consistently on several nodes, and these functions, if necessary, as a result of the self-organization of the network, can be dynamically reassigned to other nodes.

Note that in the absence of centralized control, sensor nodes can detect and correct some errors in their work, change their configuration to optimize their work and also transmit information with minimal human participation. Self-organization and decentralization also ensure network reliability and resistance to disruptions in the operation of individual nodes. While in a centralized network, a failure of the control node can lead to the impossibility of further WSN operation, in a decentralized network, in the case of a failure, the network can continue to work, since the functions of the control node can be dynamically transferred to another suitable node.

Such networks can be used actively, for example, in a disaster zone, to search quickly for people and monitor the presence of hazardous substances, or in industry and healthcare, to monitor the conditions of patients, collect medical data and automatically search for specialists in an enterprise. However, the properties of self-organization and decentralization, in addition to expanding the functionality of a WSN, can be a source of vulnerability, which an attacker can take advantage of. For instance, it could be malicious to add new nodes to the network or transfer control functions to an illegitimate node. In the case of a WSN used within a critical information infrastructure, increased attention should be paid to the security of such a network, especially if the WSN is self-organizing. More specifically, the self-organization property allows an attacker to perform various actions in the network legally, which would obviously be considered unauthorized in the absence of this property. Examples of such actions are the physical disconnection of a node from the network, dynamically adding a new node, rebuilding the network topology, changing data forwarding routes and changing the node's communication behavior pattern.

An example of a WSN within critical infrastructure is an emergency response system in a smart city. It represents a crisis response and management system targeting the mitigation of the consequences of natural disasters or other emergency situations. Thus, the possibilities of network self-organization impose additional risks associated with the unauthorized use and modification of such networks and their services by both external and internal violators of cyber–physical security. Attacks aimed at such emergency systems, if successful, can lead to serious and even catastrophic consequences of a man-made and social nature, as well as significant financial damage.

The decentralization of a WSN can be implemented using role-based functioning principles. This means that each node plays a specific role in the network; that is, it operates in accordance with the functions assigned to it [1]. Moreover, the distribution of roles between the nodes is dynamic and can change in accordance with the current operating conditions of the WSN. For example, a WSN node with a base station role fulfills the functions of processing, aggregating and analyzing primary data appearing in the network. The functions of such a node can be distributed over a few nodes and over time, if necessary; after some time passes, these functions can be redistributed to other nodes. In addition, this role-based approach makes it possible to improve the energy and resource

consumption indicators of network nodes, as the network becomes more adaptive to the risks of disruption of individual nodes.

Thus, the properties of self-organization and decentralization can be used by an attacker to perform attacks. For example, it could be an embedment of a false node into the network through using the property of self-organization, and it could lead to the distortion or substitution of critical data in the WSN. Another example is the implementation of an energy resource depletion attack by illegally exploiting the decentralization mechanism and role-based functioning of the network. All this confirms the importance and relevance of timely detection of attacking influences in self-organizing decentralized WSNs.

The core contribution of this article lies in the proposed WSN model, attack model, algorithms for data collection and attack detection, as well as the technique for applying the constructed models and algorithms. Namely, the contribution is presented by the following.

- The WSN model specifies the processes of WSN functioning by using the role-based presentation of network nodes and a specific set of their states, considering the decentralization and self-organization. In addition to that, we introduced an original set of roles, which were obtained through generalizing known possible roles described in various published papers in the field. The proposed WSN model differs from other existing alternatives in the complexity and interconnectedness of the used roles of nodes in accordance with the needs for the operation and maintenance of a typical WSN for critical scenarios and infrastructures. Nevertheless, the introduced set of roles is regarded as quite a universal one, since, unlike existing works, it does not imply linking the network to any very specific practical application area.
- A distinctive feature of the proposed attack model is that it relies on the properties of decentralization and self-organization of attacks, and these properties are considered insufficiently in the framework of other studies that have been published in the field.
- The data collection algorithm is characterized by the decentralized nature of the collection process, which can be adapted during the operation, depending on the current composition of the network nodes. The detection algorithm differs in a constructed set of informational features, which can be used to detect attacks with the use of machine learning (ML) and statistical methods.
- The main distinctive feature of the proposed technique lies in its complexity and adaptability in terms of setting the parameters of its steps to produce a software component that allows for detecting attacks in WSNs with high detection quality.

The rest of this article is organized as follows. Section 2 presents an analysis of the relevant literature in the field, including role-based WSNs, their application in critical infrastructure facilities and attack detection. Section 3 comprises the process of modeling a self-organizing decentralized wireless sensor network with role-based functioning of nodes for the sake of solving the attack detection issues. Section 4 discloses the essence and peculiarities of the attack model on a self-organizing decentralized WSN. Section 5 presents the proposed data collection algorithm. The detection algorithm is exposed in Section 6, while Section 7 presents the application technique. Section 8 embodies a description of the experiments performed and the main conclusions of the work.

2. Related Work

Information security issues in self-organizing decentralized WSNs as well as attack detection problems appear to be highly relevant in many application fields due to the spreading of such WSNs and their subjection to general and specific attacks exploiting the self-organization and decentralization. Such networks involve dynamic changing of the nodes and devices, situationally building a network topology and using various role models for the nodes assigned and redefined during the network operation. Therefore, they are currently becoming increasingly widespread. A typical example of such a WSN is an anti-crisis management system in emergencies, providing the network connectivity and communications of the heterogeneous devices related to various rescue services. Figure 1 expresses an example of a WSN as an element of an anti-crisis response system. It assumes

there are many sensor devices belonging to various emergency services. Each device takes the required readings in the emergency area and wirelessly transmits these data to other nodes and the operators. In Figure 1, different shapes of the vertices indicate different types of emergency services.

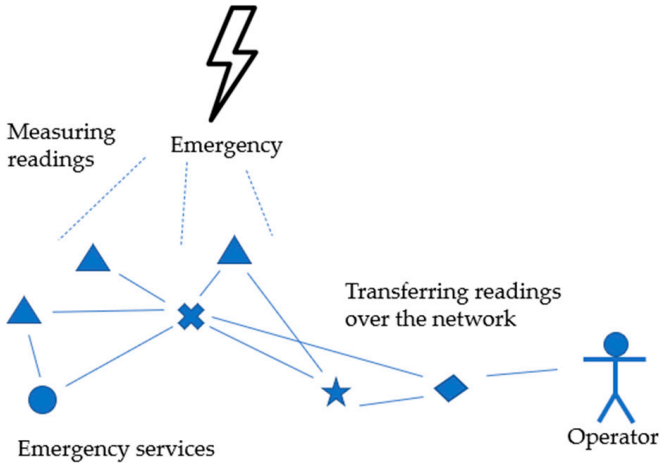

Figure 1. An example of a WSN as the element of an anti-crisis response system in emergencies.

In [2], an analysis of the use of WSN and Internet of Things (IoT) technologies is made to combat natural disasters. Examples of the use of a WSN for landslide detection, as well as vibration detection, are given. A specific WSN was used for detecting vibrations, and accelerometers were used as sensors. Durrani et al. present an overview of methods for disaster response and search-and-rescue operations by using specialized sensors in WSNs [3]. In addition, Durrani et al. note that the use of a WSN plays an important role at various stages of the disaster response, especially in the search for survivors. This circumstance could be explained by the fact that sensor nodes can function autonomously and in various conditions; for example, in open air or in destroyed places, they are also able to quickly transmit information via a wireless channel. Erdelj et al. [4] describe a disaster management system, which includes a WSN and unmanned aerial vehicle devices. The combined use of both technologies makes it possible to achieve better results in the field of disaster management. Particularly, it facilitates increasing the mobility of sensor nodes in the WSN operating area. In addition, the following works that studied the topic of WSN role functioning are considered.

Kochal et al. describe the development of the middleware for wireless sensor networks based on a model with the specified node roles [5]. These researchers note that the use of a role model can improve the efficiency of the energy and computing resource consumption during the work. In this case, the software architecture consists of three levels. Namely, these are the control level (network state and monitoring tools), the interface level (interaction of nodes with each other and external devices) and the functionality level (data collection, processing and analysis). The roles are divided into general roles and specific ones, that is, for general-purpose devices and specific types of devices, respectively. The proposed solutions are tested on a small example of a wireless sensor network.

Ortiz et al. explore wireless sensor networks by using a role-based control model [1]. The role model specifies the functional responsibilities of each role in the network and guarantees its more efficient (productive) operation. Ortiz et al. expose the results of measurements taken with various types of network organization, using the role-based management model. Several experiments are conducted to determine how the changing of the organization of the network node roles affects the network performance. The results of that research reveal that certain organizational schemes can be effectively used in various operating conditions of the sensor networks. For example, for some tasks, it may be effective to use a specific node as a coordinator that makes decisions about collecting and

transmitting data in the network. In addition, it is noted that a certain set of roles in the network can significantly improve its efficiency and help to reduce equipment costs and energy consumption. Hong et al. propose a system of node roles, which consists of the following roles: nodes with no specifically assigned roles, nodes with actuators, an observer node and a coordinator node [6].

Sudip and Ankur describe a scheme for organizing a role-based WSN, using the reputation of the nodes when assigning specific roles [7]. The reputation mechanism is used as the main parameter when assigning a specific role to nodes. It is calculated using some indicators, such as the number of successful data transmissions made by a particular node and the number of unsuccessful transmissions, as well as the number of lost packets. Each time a node interacts with other nodes, the quality of their work (reputation) is assessed, and this information is put into a special table. Thus, the nodes with low reputations cannot obtain any significant role in the WSN. For example, if a node has a large number of unsuccessful transmissions, then it develops a low reputation, and other nodes try not to use it to relay their data.

Ruairi and Keane explore the issues of constructing WSNs through using role models [8]. Their work involves dividing WSN nodes with some specific role into groups and then into dynamic regions, that is, joint groups. Thus, each region generates its routing model, which can be effectively used to transfer information. This approach is aimed at solving the problem of WSN optimization, which arises due to limited resources and the need to transmit large amounts of data over the network. In particular, the authors identify the following roles for nodes: router, proxy, producer, integrator and initiator. The research conducted by these authors shows that their approach can significantly improve the energy efficiency of the network and reduce the communication and computing resources required for data transmission. In addition, some important issues of the role-based functioning of WSNs are regarded by the authors of [9–11]; namely, in these articles, the authors describe schemes for constructing WSNs with assignments of various roles to nodes. However, they essentially repeat the work described earlier, so we will omit their more detailed consideration.

Silva et al. [12] describe a decentralized intrusion detection system (IDS). The main feature of this IDS is that its functions (i.e., the detection) are distributed over several network nodes. This IDS has been tested on several well-known attacks on WSNs, such as message delay, repetition, wormhole, jamming and data alteration attacks. The simulation results show that the proposed IDS is effective and quite accurate in detecting various types of simulated attacks.

Huang and Yuen [13] propose a scheme for decentralizing the data identification process in WSNs. In essence, this scheme presents a protocol for decentralized information processing in a WSN, designed to eliminate the shortcomings of the centralized approach, for example, the transmission of a large amount of data at high frequency and a large computing load on the central node of the network. The proposed scheme allows for reducing the load on the communication line between the nodes as well as reducing the load on the control (base) station.

Safaei et al. [14] describe a method for detecting anomalies in cases of highly noisy data. It is based on the use of time series and an adaptive Bayesian network approach. The anomaly detection algorithm proposed by the authors is a decentralized noise detection algorithm, which runs on each sensor node individually. In addition, the adaptive Bayes network is used as a classification algorithm to predict and identify outliers in each node.

Kohno et al. [15] consider a problem of secure decentralized data transmission, as well as a protection against attacks aimed at hacking keys for data encryption and authentication. A method is proposed for transmitting information from the nodes, and this method is resistant to such attacks. The Skipjack symmetric encryption algorithm is used. The main advantage of the proposed solutions is the use of the secret sharing scheme, which is based on solving the Lagrange polynomial.

Chinnasamy et al. describe issues of blocking flood attacks as they are highly relevant to wireless sensor networks [16]. The presented algorithm for blocking such attacks is based on the application of machine learning methods, comprising data preprocessing, model training, testing and evaluation. The solutions proposed by the authors are tested on CICIDS 2017 and CICDDDoS 2019 datasets, conducting experiments involving a range of ML methods, such as decision tree, Logistic Regression, etc. In addition, Chinnasamy et al. discuss issues of defense against such types of attacks as phishing that could be observed in a WSN on an application layer of the interaction [17]. The authors propose an approach to detect phishing links based on machine learning. Such attributes as link URL composition, IP address, domain registration length and others are used for the detection. The Random Forest method is chosen as an appropriate ML method. Also, they implement a software module that allows for classifying phishing links by the character of the URL. The authors managed to achieve a fairly high accuracy of the detection.

Yingxu et al. expose a method for detecting malicious nodes in wireless sensor networks based on a correlation analysis [18]. The detection of a false data injection attack in WSNs is considered. The main concept of the method is to analyze the retrospective data (i.e., sensor readings) of each node in the network and evaluate their correlation with the current readings. The ARIMA model is used as a prediction mechanism in that work. Moreover based on the retrospective data, a prediction model for new data is built. If the actual data of a node do not match the prediction, it is concluded that there is an attack. The Ada-Boost method is used to resolve contradictions when a large percentage of confidence is assigned to a sentence that has a low possibility. To reduce the influence of environmental interference on the detection process, the authors developed a method for calculating the correlation coefficients and weights. Experimental testing of the proposed solutions is conducted on a WSN containing 600 sensor nodes designed to collect information about fires (temperature, humidity and smoke concentration). The experimental results show that the proposed method gives a better performance than other ones (the false-positive rate is below 3%, while the false-negative rate is below 30%).

Bhardwaj et al. discuss the detection of network attacks in software-defined networks (SDNs) using a self-organizing intelligent tunable attack detection environment with deep learning [19]. Possible anomalies that occur in SDNs and their defense methods are described. The authors rely on intrusion detection systems, machine learning and deep learning methods. More specifically, Bhardwaj et al. propose a model of an intrusion detection system for SDN based on machine learning. The model implies a customizable concept of machine learning, which is designed to reduce the learning errors in the network and improve the detection accuracy. The authors use MATLAB/Python as software tools for implementing the model. In addition, a combined detection model that includes both machine learning and deep learning (DL) is also presented. Thus, ML and DL work in parallel, and their results are combined to make decisions on the intrusions.

Kumar et al. present a hierarchical method for detecting blackhole attacks [20]. This method is based on the introduction of an active trust and routing model with data verification during the routing. That is, in essence, the nodes calculate their trust coefficients. The authors use two protocols, namely data routing protocol and detection protocol to protect a WSN. The first protocol is used to check the data integrity, while the second one is used to detect attacks. The LEACH protocol is used to model the blackhole attack. Experiments on the attack detection show the high performance of the method. The evaluation is performed using the energy consumption, throughput and packet delivery ratio parameters.

Jane Nithya et al. review various attacks on WSNs and existing detection mechanisms [21]. They give a classification of attacks on a WSN, including routing attacks, data aggregation attacks, time synchronization attacks, replication attacks, Sybil attacks, etc. The core methods for detecting these attacks in WSNs are analyzed. The study covers machine learning, neural networks and clustering/trust methods. It describes the advantages and disadvantages of each one as well. In the experimental part, the authors conducted a

practical performance comparison of a hybrid node cloning attack detection mechanism with an entropy-based do-trust model and a one-way code attestation protocol. And it is revealed that the hybrid mechanism performs better than the other methods.

Sahu et al. discuss the models for detecting malicious nodes in WSNs [22], and an overview of various models designed for WSNs to detect malicious nodes is given. A malicious node detection scheme is proposed. It consists of specific feature extraction and modular classification. For each node, features characterizing the behavior of this node are extracted. Next, after a neural network is trained, it classifies nodes into trusted nodes and malicious ones.

These discussed works on attack detection in WSNs disclose different approaches to the decentralized attack detection methods embedded directly in the security mechanisms of the network. However, in these works, the detection processes do not explicitly consider the features of the decentralization and self-organization of the network and do not clarify how these features can be maliciously used by an attacker. In addition, some works consider attacks that could be highly relevant to WSNs in a specific context such as a phishing attack.

Since crisis response systems in emergencies and natural disasters are of a critical nature, in such systems, the issues of security of WSNs and the services they provide in the context of the actions of attackers become critically important. The mechanisms of self-organization and decentralization ensure the dynamic nature of the WSN operation. It presumes restructuring of the functions according to the current capabilities of the network throughout its functioning and the possibility of its scaling. Therefore, all of this is an advantage of the use of such networks for emergency cases. However, as we stated above, these mechanisms, in turn, create vulnerabilities, and their exploitation can disrupt the correct functioning of the WSN and lead to undesirable consequences.

We also observe that most of the related works in the field of the role-based functioning of WSNs do not provide detailed descriptions of any specific roles and their functionality. In addition, we found no published papers on the possibility of expanding roles, depending on the goals of concrete scenarios. In particular, we did not find anywhere before the possibility of introducing a special role for an attack detector in a WSN. Generally, the roles described in the published literature reflect mainly particular role-based aspects, for example, the allocation of roles in accordance with routing functions (e.g., router and proxy roles). At the same time, the roles that could specify certain peculiarities of WSN functioning, for instance, the distribution of the base station functions among other network nodes, are still uncovered in the published papers. Moreover, often, a paper refers to one specific area of a WSN application only, such as monitoring environmental conditions at industrial facilities [1].

Table 1 summarizes the attack detection results shown by the authors in the alternative works. But, it is worth noting that the relevant literature considers common variations in these attacks on WSNs, which generally do not take into account the properties of decentralization and self-organization in the process of attack fulfillment. In the case of performing these attacks in a self-organizing decentralized WSN, the attacker additionally has access to actions that are not available in the common implementation. For example, it could be the use of the role distribution mechanism to be able to obtain a large amount of network traffic. In a self-organizing decentralized WSN, sending a large amount of network traffic to a single node without its further retransmission is not always an obligatory attack. Nevertheless, this fact could be an important feature of a blackhole attack in a common WSN.

Table 1. Comparison with state of the art.

Source	Attack	Metrics	Value
Nithya et al. [21]	clone attack	precicion, recall	0.97 0.98
Chinnasamy et al. [16]	DDoS (various kinds of flood)	accuracy	0.997%
Kumar et al. [20]	blackhole	energy consumption, throughput, packet delivery ratio (PDR)	0.8 0.78–0.93 (depending on the number of nodes) 0.85–0.9 (depending on the number of nodes)
Yingxu et al. [18]	identifying malicious nodes	recall FPR FNR	0.88–0.99 (depending on the number of malicious nodes, the more there are, the lower) 0–0.03 0–0.15
Safaei et al. [14]	anomalies in WSN (Intel Laboratory dataset)	accuracy	89.9%
Silva et al. [12]	message delay, repetition, wormhole, jamming, data alteration attacks, blackhole, selective forwarding attack	detection effectiveness (accuracy)	0.9 (dalay) 0.95 (repetition) 1.0 (wormhole) 1.0 (jamming) 0.8 (data alteration) 0.96 (blackhole) 0.83 (selective forwarding)

3. Model of WSN

The proposed model of a wireless sensor network is represented by a graph of states and transitions. The necessity to develop the model follows from the need to formalize the WSN functioning processes and, in particular, to obtain a more formal view of the properties of the self-organization and decentralization inherent in such WSNs.

One way to implement the decentralization mechanism in a WSN is to introduce a role-functioning mechanism into it. Essentially, the role of a node represents a set of functions of a certain type. According to some initial distribution of the roles among the nodes, a particular node has a few functional duties assigned. However, during the network operation, the distribution of the roles among the nodes can change dynamically. For example, if needed, it is possible to redistribute the functions of a single WSN base station to some other network nodes in order to reduce the negative effect of this node when it leaves the network. Specifically, Dasgupta et al. identify some functions inherent in a base station node and sensor nodes of the network, namely data collection, sending data to a base station, data processing, data aggregation and storage [10]. Consequently, in a network involving a full cycle of the processes for collecting, manipulating and analyzing data from sensors, it is necessary to allocate the following roles: data collector (in fact, this role could be valid for almost every sensor node), data processor, data keeper and data analyzer. At the same time, typically, each role is assigned to at least one of the network nodes, and, if necessary, the role can be reassigned to another one during the operation. Thus, due to the use of the role redistribution mechanism, the runtime decentralization of the WSN operation is ensured.

A detailed description of the characteristics of each role in the WSN model we proposed and a description of its functionality are presented in Table 2. The system of the roles is specified as a set

$$R = \{r_{coll}, r_{keep}, r_{proc}, r_{control}\}, \quad (1)$$

where R is a set of roles used to organize the decentralized functioning of the WSN, r_{coll} means the data collector role, r_{keep} is data keeper role, r_{proc} is data processor role, $r_{control}$ is network controller role.

Table 2. Description of roles in a self-organizing decentralized WSN.

Role	Role Characteristic
Data Collector r_{coll}	Collect data from sensors and user interface elements connected to the node. The initial nodal preprocessing of data and its transformation to a unified form is performed. This role is given to each WSN sensor node by default, regardless of whether the node has another role (assuming each normal node collects and preprocesses its readings from its sensors). The role could be taken off from a node if its sensor readings are no longer needed. After the preprocessing, typically the data is transferred to the nodes with the role of data keeper.
Data Keeper r_{keep}	It stores data from all collector nodes and provides access to this storage for the nodes with data processor roles. This role can be assigned to dependable nodes with a large amount of storage. This role can be removed from a node in case the node runs out of storage memory or if low remaining energy resource of the node or in case some new node being more advantageous for this role is added to the network. In the latter case, the network controller makes a decision about the roles redistribution.
Data Processor r_{proc}	Processing and aggregation of coming data and historical data for certain periods. This role could be assigned to the nodes with large computing resources. The decision to reassign this role to another node could be made, if the current one is not capable of performing its functionality due to the low levels of energy or computing resources, or if a node being more advantageous for this role is appears in the network.
Network Controller $r_{control}$	It makes decisions on network reorganization and redistribution of the roles based on its network management model. The network controller provides service messages for other nodes. This role is assigned at the network initialization stage and can be reassigned if the current node with this role is not capable of performing its functionality, for instance, due to the low energy or computing resources. If the node with this role suddenly leaves the network, the rest nodes are forced to reinitiate the network by using the cooperative control protocol and choosing a new node for the Network Controller role.

The initial distribution of roles among WSN nodes is performed at the WSN initialization stage. By default, each node could be assigned the data collector role. Further, during the operation of the network, the assignment of roles can be redistributed in accordance with the conditions described in Table 2. Thus, after a new node appears in the WSN, the network controller makes a decision upon assigning some specific role to this node if it is more profitable for the functioning of this role.

The model of a self-organizing decentralized WSN with role-based functioning is represented by a graph of states and transitions shown in Figure 2. The set of vertices of the graph is specified as follows

$$S = \{S_0, S_1, S_{s2}, S_{s3}, S_{s4}, S_{s5}, S_{s6}, S_{d2}, S_{d3}, S_{d4}, S_{d5}, S_{d6}, S_{d7}\}, \quad (2)$$

- where S_0—initialization state of WSN,
- S_1—state of WSN normal functioning. A transition to the next state S_{s2} occurs if a new unregistered node appears in the network. A transition to the next state S_{d2} is performed if some node cannot fulfill its assigned role (for example, the resources are running out) or when the number of nodes in the WSN changes.
- S_{s2}—state of adding a new node to the WSN. After receiving this request from a new network node, the transition to the next state occurs.
- S_{s3}—state of waiting for a request to add a new node to the network. After receiving this request from a new network node the transition to the next state occurs.
- S_{s4}—state of changing the set of active WSN nodes. The transition to state S_{s6} occurs after the successful confirmation of a new node for inclusion in the WSN (S_{s5}).
- S_{s5}—state of waiting for confirmation of a new node inclusion in the WSN. The new node sends confirmation of joining the network and transitions back to the state S_{s4}.
- S_{s6}—state of redistribution of WSN roles. If a new node's characteristics are suitable for running a specific role (data keeper, data processor or data collector) and surpass

the current node for this role; then, the mechanism for redistributing roles in the network could be launched and the transition to the state S_1 initiated.

- S_{d2}—state of requesting for redistribution of roles in the WSN. After receiving and processing the request by other nodes, the network passes a transition to the state S_{d3}.
- S_{d3}—state of changing roles of WSN nodes.
- S_{d4}—state of updating the current role distribution of the WSN. This state presumes to send information to the network controller about the role of each WSN node and the current characteristics of the nodes (including communication and computing resources, energy resources and readiness to accept a specific role). After that, it makes a transition to state S_{d3}.
- S_{d5}—state of notifying nodes about role redistribution. This state indicates broadcasting a service message containing the new roles assigned to the nodes. The transition is made to state S_{d3}.
- S_{d6}—state of waiting for confirmation of a new role assignment. The state indicates sending confirmation by each node of readiness to function in accordance with the newly assigned role to the coordinator. The transition is made to state S_{d3}.
- S_{d7}—state of updating the role distribution between WSN nodes. Each node updates the table of roles of other nodes and its role. The transition is made to state S_1.

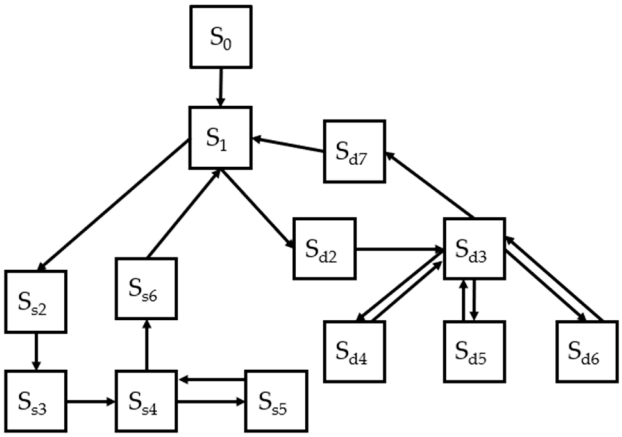

Figure 2. State and transition graph illustrating the states of a self-organizing decentralized WSN and transitions between them.

In the state symbols, the indices 's' and 'd' denote the connection of the state with the properties of self-organization and decentralization of the WSN, respectively. For example, state S_{d2} determines the performance of the procedure for generating a request for redistribution of roles, thereby providing decentralization, whereas state S_{s2} presents the addition of a new node, providing the property of self-organization.

In the graph shown in Figure 2, the arcs indicate transitions between specific states of the WSN. The fact of each transition is determined by running several actions at certain nodes of the WSN and triggering certain conditions. The graph of states and transitions described in Figure 2 exposes the operation of a self-organizing decentralized WSN as a whole. In Figure 2, the arc arrows show the direction of the corresponding transitions between the states. Next, let us describe the state model that is inherent in each specific WSN node. The model is presented in the form of a graph of transitions and states. This model is specified by the following set:

$$\text{Snode} = \{S_0, S_{coll}, S_{proc}, S_{keep}, S_{control}\}, \quad (3)$$

- where S_0 denotes the state of initialization of a WSN node,
- S_{coll}—state of WSN node functioning with the role of data collector. A transition to one of the following states is possible: S_{proc}, S_{keep}, $S_{control}$. The transition is performed if a decision is made to redistribute the roles of the WSN nodes.

- S_{proc}—state of WSN node functioning with the role of data processor. A transition to one of the following states is possible: S_{coll}, S_{keep} and $S_{control}$.
- S_{keep}—state of WSN node functioning with the role of data keeper. A transition to one of the following states is possible: S_{coll}, S_{proc} and $S_{control}$.
- $S_{control}$—state of WSN node functioning with the role of network controller. A transition to one of the following states is possible: S_{coll}, S_{proc} and S_{keepl}. The transition is performed if a decision is made to redistribute the roles of the WSN nodes. The transition is passed to a state corresponding to the newly assigned role.

An illustration of the graph of possible node states in a self-organizing decentralized WSN is presented in Figure 3. The arrows expose the directions of the transitions between the corresponding states.

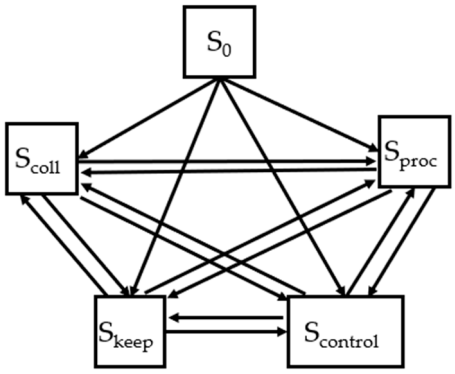

Figure 3. State and transition graph illustrating possible states of a WSN node.

In the graph shown in Figure 3, the arcs indicate transitions between specific states of the WSN node. And the fact of each transition is determined by running a number of actions and triggering certain conditions. Thus, the model of a self-organizing decentralized WSN with role-based functioning is represented in the form of two graphs with states and transitions. The graphs determine the process of functioning of the network as a whole and the process of functioning of each specific network node.

4. Attack Model

In order to analyze the most relevant security threats for the class of networks under consideration, a threat model for a self-organized decentralized WSN was made. This model is intended to help identify the most important threats and, consequently, attacks that need to be detected in such a network. The threat model is presented in Table 3.

Table 3. Threat model of a self-organized decentralized WSN.

Treat	Importance of the Threat	Probability of Realization	Description and Examples
Data integrity breach	high	high	Modification of data transmitted over the network. The modified data can distort the parameters measured by nodes or change the parameters of WSN control commands (e.g., Sinkhole, Wormhole)
Data availability breach	middle	high	Disconnecting nodes from the network or creating conditions where nodes cannot respond to requests in a reasonable amount of time. It causes data loss in the WSN. (e.g., jamming, DoS, blackhole)
Data privacy breach	middle	high	Copying of data transmitted over the network by an attacker. It can be used in preparation for other attacks. (e.g., sinkhole, Sybil, wormhole, blackhole)

Table 3. *Cont.*

Treat	Importance of the Threat	Probability of Realization	Description and Examples
Disruption of routing in the network	high	high	Forced modification of routing paths in WSNs. Disrupts normal WSN operation. (e.g., Sinkhole, Sybil, Wormhole, Blackhole)
Communication protocol violation	high	middle	Changes in the logic of the decentralized functioning protocol leading to loss of WSN operability. (e.g., desynchronization, clock-skewing and data-replay attacks)
Violation of the algorithm of nodes operation	high	middle	Modification of software on the nodes. It leads to violation of algorithms of nodes operation and WSN as a whole (e.g., malicious code injection)

After building the WSN model and tread model, we specify the attacks that a self-organized decentralized wireless sensor network is subject to. Generally, the attack model is represented as a set of tuples:

$$\{(\text{Type}_{\text{impact}}, \text{Type}_{\text{attack}}, \text{Target}_{\text{attack}}, \text{Involv}_{\text{selforg}}, \text{Involv}_{\text{decentral}})\}, \quad (4)$$

where $\text{Type}_{\text{impact}}$ specifies the subclass of the object that is affected. More specifically, $\text{Type}_{\text{impact}}$ specifies data, software of the nodes, network and transport layer protocol used or the physical communication channel. $\text{Type}_{\text{attack}}$ determines a particular type of attack, i.e., the method used to influence the target of the attack. $\text{Target}_{\text{attack}}$ specifies the particular target of the attack. $\text{Involv}_{\text{selforg}}$ represents a binary characteristic determining if the attack is based on the exploitation of the self-organization property. Similarly, $\text{Involv}_{\text{decentral}}$ represents a binary characteristic determining if the attack uses the exploitation of the decentralization property. Thus, each specific instance of the tuple defines a set of five characteristics of the attack under consideration, while the model presents a list of such tuples. As an example for a wormhole attack [16], the tuple takes on the following form: data transmitted to the WSN, wormhole type, violation of confidentiality and data integrity, true, true.

Let us consider the following main and most important types of attacks inherent in WSNs. The essence of a sinkhole attack, which aims to both eavesdrop on network data and modify them, is that a malicious node forces other nodes to transmit data through it and, thereby, 'convinces' them that data transmission through it represents the shortest route [23,24]. Therefore the nodes begin to transmit data through this attacking (infected) node only. The attack disrupts the routing mechanism and can be performed by introducing a new node into the WSN or by influencing an existing one.

A wormhole attack intercepts a message from one node to another and delivers it faster, that is, in fewer hops. As a result, the original, legitimate copy of the message will be discarded by the recipient node [23,25,26]. This attack can be mounted using two malicious nodes that can transmit messages to each other over a communication channel different from the channel the WSN uses. Note that this attack is also aimed at disrupting the routing process as a whole, and its goal is to modify and eavesdrop on data in the network.

The goal of a blackhole attack is to interrupt the communication in the network when the attacking node takes (in fact, loses) received packets, and other nodes begin to look for alternative routes in the network [23,27]. In this case, the nodes have to resend lost packets, including through other routes. The attack disrupts routing in the network, and due to the need for retransmission, there is unnecessary consumption of communication and computing resources of the network.

A Sybil attack consists of generating information on network nodes about the existence of several different nodes, which, in fact, are one attacker node. And this node also violates the routing rules in the network [23].

A malicious code injection attack is a type of attack that exploits errors in software and injects malicious code into the host control program, for example, by injecting code into a data packet [28,29]. The goal is to disrupt the operation algorithms of the node.

Data aggregation distortion and node-tampering attacks involve the introduction of malicious code to disrupt the operation of nodes. A data aggregation distortion represents an attack on a node that processes and aggregates data received from other nodes. An example of an interference attack is changing the operating algorithm of a node's control program or adjusting the physical environment of its surroundings. Such an attack can lead to distortion in the readings of its sensors.

Hello-flood attacks and Denial-of-Sleep attacks inherent in wireless sensor networks are types of DoS attacks [23,30,31]. Hello-flood attacks consist of frequent sending of 'Hello' broadcast messages, informing neighboring nodes about the proposed communication parameters. The nodes receiving such messages are forced to react to them. The receiving node can adjust its list of network nodes available for communication with it and adjust its routing tables. The attack leads to an unnecessary consumption of energy and computing resources of nodes, and, therefore, it can be quite effective in hitting WSNs. Often, a hello-flood attack involves the attacker using a high-power transmitter to hit as many network nodes as possible and, thereby, increase the volume of information flows in the network. The target of the Denial-of-Sleep attack is to prevent a node operating from an autonomous power source in the network from switching to energy-saving mode. This influence leads to an increase in energy consumption of the victim node. For instance, the attack can be performed by sending a hello request and can lead to a loss of the node availability.

Desynchronization attacks, clock-skewing and data-replay attacks can be classified as attacks on the protocol of interaction between nodes [23,32]. A desynchronization attack involves sending packets with a duplicate of the current session number. Since, in accordance with the rules for the correct functioning of the network, there should not be two identical session numbers at the same time, the attack leads to the closure of the session with the sent sequence number. Therefore, such nodes have to create new sessions (and this, in turn, implies preliminary message exchange), which additionally consumes node resources and, as a result, can lead to disruptions in the functioning of the network. A clock-skewing attack is an attack on WSN sensors that require synchronization of the current time to operate. It desynchronizes the sensors, spreading false values of the current time moment. A data-replay attack involves an unauthorized repetition of data packets. An attacker resends correct, previously recorded data packets over the network. This action can lead to distorted information about the state of the environment where the WSN operates.

A jamming attack consists of noising the communication channel through which WSN nodes communicate [33,34]. An attack can be classified as a physical method of influence. The attack can lead to a complete disruption of the network or the disconnection of one, two or more nodes in the network. The attack is harder to detect and may not be immediately revealed by an attack detection mechanism due to the presence of self-organizing properties in the network.

Figure 4 shows an example of the use of self-organized decentralized WSNs as part of a smart city system. In the figure, the arrows show the connections between specific WSN nodes that define the network topology at a certain point in time. The elements of the figure indicate nodes that measure ambient air parameters (temperature, humidity, etc.). Some of them are separate devices, while others are installed on elements of the city's infrastructure. In this example, the WSN nodes monitor environmental parameters (temperature, humidity, pollution, etc.) and are located on both individual devices and vehicles. In case a WSN node is a separate device, it is periodically moved from one point in the city to another. Self-organization allows the nodes to move freely in space so, at the same time, the performance of the network is not disturbed, because when one node leaves, the intended messages will be transmitted through the other nodes. Decentralization allows one to distribute the processing, analysis and storage of information among several WSN nodes, while not overloading any node with these computational processes. Role-based

functioning keeps the network operation in the case of disconnection of one of the nodes from the network (i.e., the role performed is passed to another node). It also allows one to balance the performance of the network if a new node is added to it (i.e., the new node starts realizing a specific role while reducing the load on the other nodes).

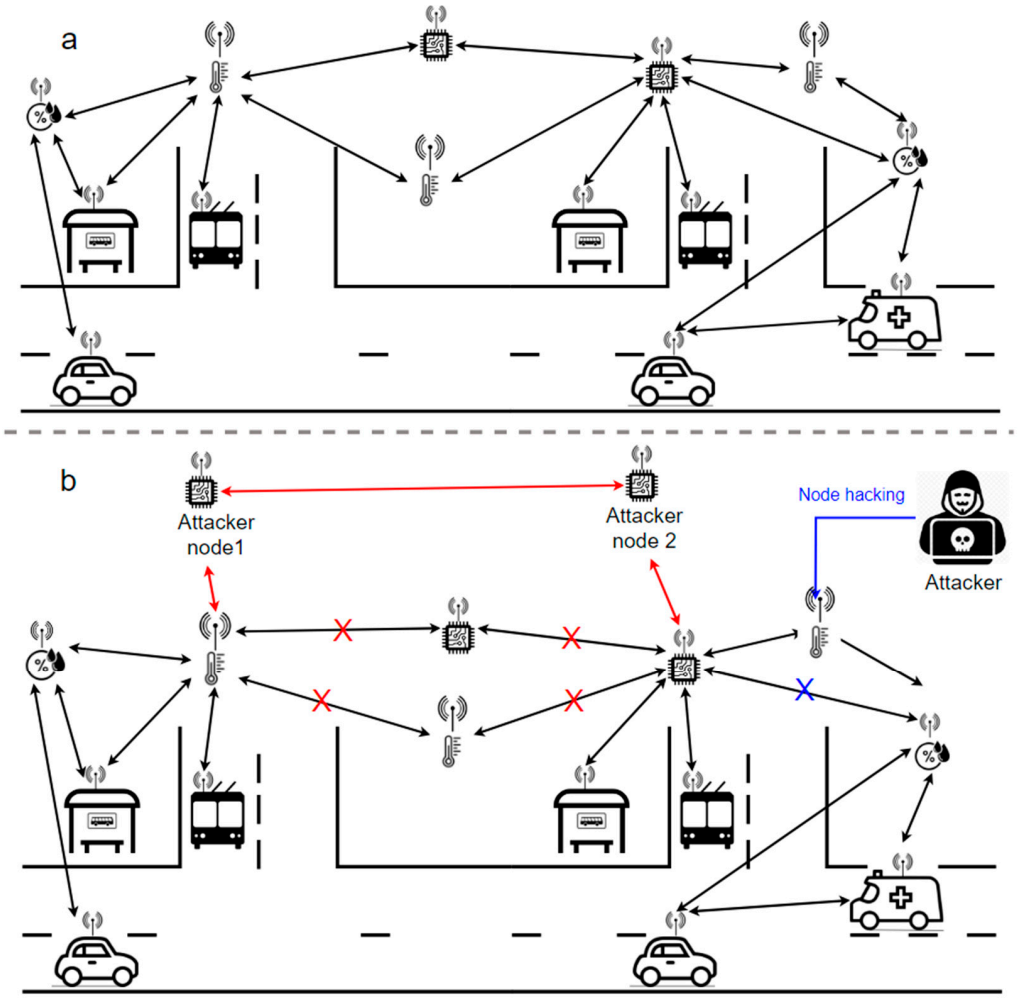

Figure 4. Case study of air pollution monitoring example (in an urban infrastructure). (**a**) Initial WSN indicating the connection between the nodes; (**b**) WSN under the influence of wormhole and sinkhole attacks.

Specifically, Figure 4a discloses the initial state of WSN, where there are two subnets connected to each other through two specific nodes. Data can be transmitted seamlessly through one of the possible routes from one part of the WSN to the other. Part b of Figure 4 illustrates two attacks. Namely, a wormhole attack is presented in red and, a sinkhole attack is in blue. The wormhole attack exploits the self-organization property to introduce two malicious nodes into the network, disrupting the routing processes of the network. The decentralization property, in practice, can exacerbate the effects of the attack, as the added nodes can be given specific roles and the WSN operation will be disrupted. Eventually, data from one part of the WSN to another are transmitted through the attacker nodes, while the data transmitted over the legitimate communication channel are discarded. The attacker nodes can eavesdrop or modify information, compromising data confidentiality and integrity. In addition, in such cases, the routing in WSNs is disrupted. The goal of the sinkhole attack is to redirect all traffic in WSN through a single node controlled by the attacker. In addition, Figure 4b shows in blue that the attacker compromises one of the nodes in the WSN and eliminates all routes in the network, leaving only one route through the controlled node. This attack can also be realized through adding a new node to the

WSN and result in compromising data confidentiality and integrity as well as disrupting the routing in the network. In addition, the controlled node in the network can be given any role, and an attacker can interfere with the implementation of that role.

In the works mentioned above, their authors mostly consider the simplest (i.e., common) versions of these attacks, that is, variants of the attacks without taking into account the property of decentralization of the network. However, self-organization and, to a greater extent, decentralization may allow an attacker to mount an attack less noticeable to security means than in its common variation. An example of a more covert variation of the attack is an exploitation of decentralization and role distribution in order to illegitimately obtain a specific role to learn any extra information from other network nodes. Another example is to exploit the role redistribution mechanism to mount a Denial-of-Service (DoS) attack maliciously, namely a flood attack, wasting the available energy resources of the nodes. All this allows us to conclude that the analysis of the unauthorized influences of self-organization and decentralization properties on the attacks described above is highly relevant and effective and, hence, it is an important task. Therefore, this implies the need to develop algorithms for the detection of such attacks as crucial steps to counter these influences.

As stated above, the considered attacks on self-organizing decentralized WSNs can be divided into several subclasses, characterizing the object they influence. Namely, these can be attacks on data transmitted over the network; impacts on node software; influences on physical part devices of WSN nodes; and impacts on the interaction protocol and communication channel. Summarizing for each class of attacks, possible types of attacks are given in Table 4.

Note that the attacks given in Table 2 are feasible both in common WSNs and in self-organizing decentralized WSNs. However, after a WSN becomes one with properties of self-organization and decentralization, the attacker finds expanded opportunities to perform these attacks by using actions that are a priori considered as legitimate ones in such networks. For example, in a common WSN with a fixed structure, the appearance of a new node in the network usually presents a clear feature of a malicious modification of the network structure, while in a self-organizing network, this fact, in itself, cannot directly indicate any attack. Therefore, to detect attacks on self-organizing decentralized WSNs, an expanded set of informational features and more complex methods for their detection are required.

For each of the attacks considered in the work, we analyzed to what extent the properties of self-organization and decentralization, as well as the role-based functioning of the network, can influence the feasibility of the attacks. It was revealed that in some cases, e.g., for a sinkhole attack, the realization of the attack is simplified, while for other types of attacks, for example, a data-replay attack, on the contrary, it becomes more complicated. The results of the analysis are presented in Table 4. The table specifies possible types of attacks on this class of networks, disclosing the distinctive features of each attack. Also, Table 3 exposes the results of an analysis of the influence of the properties of self-organization and decentralization on the feasibility of the considered attacks. Thus, the advantage of this attack model is to infer the most important types of attacks that WSNs are subject to, as well as the main characteristics of such attacks to be used as the basis for a feature space construction for the further detection of these attacks.

Note that the attack model described in Table 4 is not exhaustive and could be supplemented by some other types of attacks, but at the same time, it covers all the main, most important typical types of attacks described in the literature. For example, the attack of physical destruction of a WSN node is also relevant for the considered network. It is not included in the model due to its low efficiency and probability of realization. As another example, flooding attacks such as UDP-flood or ICMP-flood may also be relevant in self-organized decentralized WSNs if they use these protocols. However, note that the considered WSN does not use UDP and ICMP protocols, so these attacks are not included in the model.

Based on the analysis, it can be deduced that the realization of some types of attacks, for example, sinkhole or DoS, can be simplified due to exploitation of the properties of self-organization and decentralization. However, such an attack detection is supposed to be complicated, since the impact is similar to the processes of the normal WSN functioning. Therefore, it is highly important to develop detection algorithms that will allow for timely detection of such attacks.

Table 4. Analysis of attacks considering the use of self-organization and decentralization properties.

Attack Types	Target	Impact of WSN Self-Organization	Impact of WSN Decentralization
1. Impacts on user and service data in WSN			
Sinkhole, Sybil, wormhole, and blackhole	Violation of the integrity and confidentiality of transmitted data (including routing data)	Expanded attacker capabilities to illegally intrude a node into the network, as well as to spoof node addresses	Exploitation of the functions of assignment and redistribution of node roles by an attacker
2. Impact on WSN software			
Malicious code injection and data aggregation distortion	Modification of software algorithms running on WSN nodes	Expanded attacker capabilities to illegally intrude a node into the network and rebuild the logical structure of the network	Infecting a node with a modified command to change the role of the node.Forced acquisition of data keeper role or data processor role
3. Impacts on physical part of WSN nodes			
Hello-flood attack, Denial-of-Sleep, and node tampering attack	Violation of the availability of nodes, leading to disruption of the operation, incl. depletion of energy resources of WSN nodes	Possibility of intrudinginto the network a significant number of new nodes with given network settings. These nodes are difficult to be classified a priori as anomalous ones	Possibility of exploiting the weak points of the decentralization mechanism, incl. assigning roles to non-existent nodes
4. Impact on the WSN communication protocol			
Desynchronization, clock-skeing and data-replay attacks	Distortion of the node interaction protocol	Possibility of authorized addition of an attacker node to the WSN	Exploitation of protocol weaknesses, incl. modification of protocol fields 'on-the-fly'
5. Impacts on WSN link channel			
Jamming	Jamming of communication channels through which wireless communication is performed in the network	Due to the normal self-organization of the WSN, the attack can be recognized as a legitimate change in the WSN	Possibility of influencing nodes and initiating the process of role distribution involving all WSN nodes

5. Data Collection Algorithm

To develop algorithms for detecting the attacks described in Sections 2 and 4, it is necessary to specify the initial data that need to be collected on network devices. Particularly, here, we need to collect data to form possible features of attacks and data about their source, i.e., the network nodes initiating the attack. Table 5 discloses the types of data that need to be collected to detect attacks described in the attack model.

Table 5 shows the relevant types of data for the attacks, where the intruder exploits the properties of self-organization and decentralization. Note that these attacks could be detected with difficulty by attack detection tools used for common WSNs.

The considered WSNs are of role-based functioning, including various functions, such as collecting, processing and storing information, to be assigned to various network nodes. Detection of attacks in such a WSN should be performed using a special node with a new detector role, and all developed algorithms for data collection and attack detection should be implemented by this node as well.

Table 5. Types of data collected for attack detection.

Data Type	Source of Data	Unit	Attack Examples
WSN message routes	Nodes routing tables	pcs.	Sinkhole, Sybil, wormhole, blackhole
Requests to redistribute nodes roles	Network Controller service log	event and them attributes	Sinkhole, Sybil, wormhole, blackhole
Nodes sensor readings	All sensor nodes	fact of reading	Malicious code injection
Control software checksums	All nodes	value	Malicious code injection, data aggregation distortion
Payload size of data packets	Message packets transmitted over the network	bytes	Malicious code injection
Node resource consumption (CPU load, memory consumption, remaining energy resources)	All nodes	percent	Denial-of-Sleep, hello-flood
Geolocation of nodes	All nodes	GPS coordinates	Jamming, sinkhole
Events of adding new nodes to the network and nodes leaving from the network	Network Controller service log	event and its attributed	Jamming, sinkhole, Sybil, wormhole, blackhole
Wireless signal level	All nodes	dBm	Jamming

The proposed combined algorithm for data collection in a WSN contains six main steps. The algorithm is presented in Figure 5. The distinctive differences of the algorithm present the specially selected types of needed data, as well as identifying information on their sources.

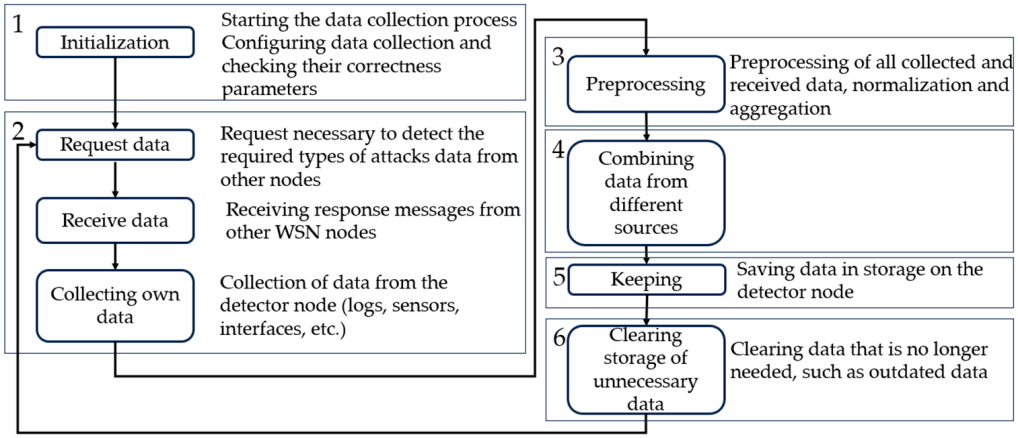

Figure 5. Combined data collection algorithm for detecting attacks in WSNs.

Step 1 involves the initial configuration of data collection parameters. The interval from the data request, the types of data collected and the sources for obtaining them are specified. For example, for data structures, 'requests for redistribution of node roles', the network controller node is requested. The correctness of the specified parameters is also checked. Step 2 involves directly requesting data from other network nodes, waiting for a response from them, as well as collecting data (i.e., detector logs).

Steps 3 and 4 involve preliminary processing of the obtained data, as well as their unification, that is, bringing data from different sources to a single form.

Steps 5 and 6 involve storing data in the detector node's storage and periodically clearing the outdated data, respectively.

Algorithm 1 presents a pseudocode of the developed data collection algorithm. As input, this algorithm has a list of all WSN nodes with their addresses and a list of required data. The output contains the data stored in the data storage, which should be used for the detection. The required data are requested from other WSN nodes, and then they are checked, preprocessed, and written into the database.

Algorithm 1. Data collection algorithm.

```
Input: list_of_nodes, nesesary_data_types
Output: collectrs data saved to database

def dataCollection(list_of_nodes, nesesary_data_types)
    chanel = Open communication chanel
    # Send request to all nodes
    for node in list_of_nodes:
        SendRequestToNode(chanel, list_of_nodes.node_address,
        nesesary_data_types)
    # Create list to save response
    nesesary_data = Empty_list
    # Delay before get response
    time.sleep(5)
    # Get response. When response stop—exit
    while True
        data = getResponseFromNodes()
    if data is None:
        break
    correct = checkCorrectOfData(data)
    if correct is True:
        nesesary_data.append(data)
    # Preprocessing
    for data in nesesary_data:
        data = Preprocessing(data)
    # Save to database
    db = openDatabaseConnection()
    db.save(nesesary_data)
    # Clear database fron old data
    if timestamp_of_data_from_db >= time_to_keep
        clearDatabaseFromOldData()
```

Thus, the algorithm described above will allow for organizing the ongoing collection of data for attack detection on the sensor network. Note that the algorithm uses a model of initial data (Table 5), which is specific to a particular network. And, therefore, in a general case, the running of the data collection algorithm should be preceded by refining and updating this model for a specific application scenario.

6. Attack Detection Algorithm

After performing the process of data collection in a self-organizing decentralized WSN, it is necessary to construct an algorithm for detecting attacks for such networks. Currently, in practice, there are a number of different algorithms for detecting attacks in various networks, including in WSNs, based on profiling attacks and normal behavior of WSNs, the use of machine learning and data mining, statistical, and other methods. Each of these algorithms can be used to ensure the security of self-organizing decentralized wireless sensor networks with role-based control.

But, in the context of the problem to be solved, the most promising seems to be a combination of several algorithms to ensure the identification of several of the most relevant attacks on the network at once. For example, a rules-based algorithm would be suitable for initial verification of the adequacy of sensor readings on nodes. To implement it, it is enough to set the required ranges of readings, and each node will be able to independently identify abnormal values. Additionally, it can be noted that the algorithm does not require large computational resources, which has a positive effect on the operation of network nodes. Thus, one can identify attacks aimed at falsifying sensor readings. To detect DoS attacks on WSNs, as a rule, a statistical approach using heuristics is used. A significant increase in the number of unsuccessful sessions or nodes connected to the network per unit of time may indicate the presence of an attack. Profiling allows for the detection of previously unknown attacks and anomalous behavior in general. The proposed combined algorithm for detecting attacks in self-organizing decentralized WSNs is presented in Figure 6. The algorithm combination is expressed in the joint usage of algorithms for different types of attacks into a single detection algorithm.

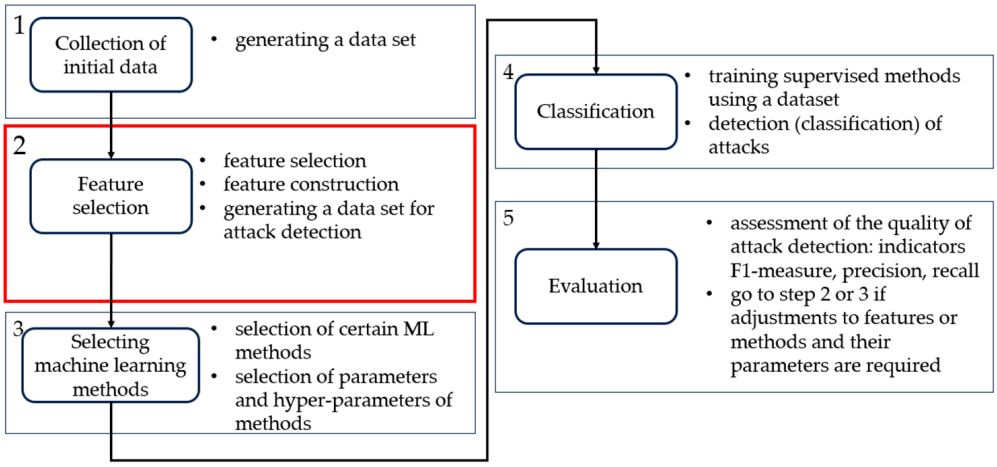

Figure 6. Combined algorithm for detecting attacks in WSNs.

The combined algorithm contains the following five steps. In step 1, a dataset is generated from previously collected data from the WSN. Step 2, which is seen as the most important one, involves selecting a feature space, calculating the features and transforming a dataset that will be suitable for detection algorithms for different types of attacks.

Step 3 involves choosing specific machine learning methods and tuning their hyper-parameters to detect the required types of attacks. Steps 4 and 5 represent training machine learning methods as well as conducting experiments on attack detection and evaluating the detection quality, respectively. If the detection quality is not acceptable, then adjustments to the feature space or learning methods should be made.

Algorithm 2 illustrates the pseudocode of the proposed attack detection algorithm. The input data contain a dataset with the data required for detection and a list of detection methods. The output data are the detected attack incidents as well as the detection quality assessment metrics. The first step is to read the required data and database; then, feature selection and the construction of new features are carried out. The last steps are model training, classification and detection quality evaluation.

Algorithm 2. Algorithm for detecting attacks.

```
Input: data from database, List of ML models
Output: detected attacks, quality metrics
def detectingAttack(list_of_features, ML_methods)
    # Read data from database
    db = openDatabaseConnection()
    data = db.readData()
    # Do feature construction from data from database
    for features in list_of_features:
        constructed_features = featureConstruction(data)

    features_for_detect = featureSelection(features, constructed_features)

    for methods in ML_methods:
        methodsSelection()
        selectionML_parameters()
    train_ds, test_ds = train_test_split(data)
    for methods in ML_methods:
        methods.train(train_ds, features_for_detect)
        attacks = methods.predict(test_ds)
        metrics = methods.evaluate(attacks, test_ds)
```

The main contribution of detection algorithms lies in the feature space for detecting specific attacks. Specific features for their detection were compiled for each class of attacks, as summarized in Table 6.

The features described in Table 6 can be used for attack detection and can be obtained from the collected data, either directly or after their statistical and other processing. For example, the feature of the number of involved routes per unit of time can be obtained from the WSN message routes, whereas the average frequency of changing the roles of network nodes can be calculated by analyzing requests for the redistribution of node roles per unit of time.

The feature selection for detecting the mentioned classes of attacks in the considered WSN class was performed through an expert analysis of the information that can be collected in the network. The specific features of the implementation of each attack were analyzed, identifying the data that change due to the impact of the attack.

For example, when the sinkhole attack is realized, the routing in WSN is changed so that all network traffic passes through the infected node. Thus, a large increase in the number of unique routes passing through a certain node per unit of time and the number of involved routes per unit of time may indicate a sinkhole attack. Therefore, by using the above features and some additional features (such as requests to roles redistribution of network nodes and the attributes of the request, attributes of an event of adding a node to the network), it is possible to detect impacts on user/service-related data in a WSN.

In addition, for optimization purposes, the feature selection is performed using a number of program functions embedded in machine learning libraries, such as the functions f_classif and mutual_info_classif, of the feature_selection class in the scikit-learn library. They allow for calculating the dependence of the target variable (i.e., attack/norm) on specific features. Using these functions, the most relevant features are selected to detect specific attacks. However, in this article, the selection of features was fulfilled primarily on the basis of analyzing the peculiarities of the implementation of attacks. Note that using features that weakly correlate with the resulting variable may disrupt machine learning techniques and result in poor detection quality.

Table 6. Features for attack detection.

Attack Class	Features
1. Impacts on user and service data in WSN	- number of involved routes per unit of time; - number of unique routes passing through a certain node per unit of time; - requests to roles redistribution of network nodes and the attributes of the request (incl. the address of the WSN node being the initiator of the role redistribution; characteristics of this node, including the amount of free memory of its local storage and the remaining energy resources); - attributes of an event of adding a node to the network (incl. geolocation and resource characteristics).
2. Impact on WSN software	- average frequency of sensor readings by nodes per unit of time; - control software checksums; - payload size of data packets; - requests to change the roles of network nodes and their attributes (ID of the node being the initiator of the request, its characteristics).
3. Impacts on the WSN node	- average frequency of roles redistribution of network nodes; - consumption of node resources per unit of time; - geolocation of nodes; - number of events of adding nodes to the network per unit of time; - frequency of nodes reading sensor readings per unit of time.
4. Impact on the WSN communication protocol	- requests to redistribution of network nodes roles and the attributes of the request (request state and count of this request); - number of events of adding nodes to the network per unit of time.
5. Impacts on the WSN link channel	- wireless signal quality; - geolocation of nodes; - number of events of nodes leaving the network per unit of time.

7. Technique for the Models and Algorithm Application

This section describes the technique for combined usage of the proposed models and algorithms in self-organizing decentralized WSNs. It is aimed at ensuring the effectiveness of attack detection in compliance with decentralized WSN architectures.

The technique relies on the use of machine learning methods and it is proposed to combine data received from different network nodes to improve the quality of attack detection. In addition, the technique uses requirements for the reliability of the attack detection algorithm as well as ones to false positives. Schematically, the proposed technique is presented in Figure 7.

The technique consists of five main stages. A list of attacks that need to be detected and the characteristics of a specific WSN form the input data of the technique. It includes network characteristics containing the properties of self-organization and decentralization, a list of node roles, types of sensors used, the purpose of the WSN, etc. The technique also takes into account a number of specific requirements and restrictions, including restrictions on the amount of hardware resources consumed by the detection algorithms, as well as the requirement to maintain the network functionality during the process of attack detection.

Stage 1 involves a specification of the WSN and the construction of an attack model. At this stage, a formal description of the WSN and its properties is made. In addition, it includes the identification of relevant types of attacks with an assessment of the influence of self-organization and decentralization properties on the course of each attack.

Stage 2 determines a collection of primary data necessary to detect attacks. Data collection parameters are configured. The location and duration of their storage and preprocessing algorithms are specified. The result of this stage is a labeled dataset intended for the detection algorithms based on supervised machine learning methods. In stage 3, these data are used to produce informational features as well as the training methods selected with suitable hyper-parameters.

Stage 4 presents a deployment of the obtained attack detection algorithms to a node with the role of an attack detector. Stage 5 includes conducting experiments to detect attacks, evaluating the quality of detection and using accuracy, precision, recall and F1-measure. Generally, stage 5 is optional and involves testing the attack detection algorithm based on previously obtained initial data, i.e., logs of the functioning of the network under attacks. The output of the technique is a software module configured to implement the processes of collecting data on the network and detecting attacks based on them that are relevant to the WSN in question. Thus, the technique allows for configuring and implementing software security modules for the effective detection of relevant attacks on a self-organizing decentralized WSN.

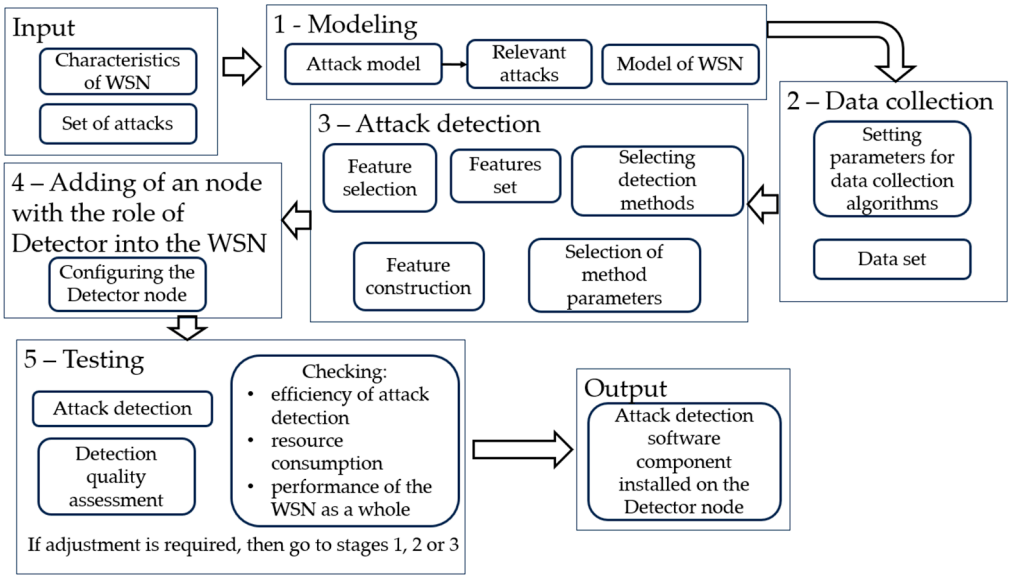

Figure 7. Technique for applying the proposed models and algorithms.

8. Experiments and Discussion

To check the correctness of the proposed models and algorithms, a hardware/software test-bench of a wireless sensor network was developed and assembled. This test-bench was partially described already in [35]. To implement self-organization of the network, ZigBee network layer protocol was used. Decentralization and role-based functioning were implemented at the application level in Python. Each network node consists of a single-board computer Raspberry Pi version 3 or 2, connected sensors (temperature, pressure, humidity, etc.) and a Digi XBee communication module [28,29]. The main characteristics of the nodes are presented in Table 7.

The role distribution is specified in a configuration file, which is replicated to all network nodes. Particularly, the configuration specifies a node identifier, types of sensors, update frequency of sensor readings and addresses of other nodes, indicating their roles. In case of a role change of a node or redistribution nodes, this file is automatically sent and adjusted consistently on all nodes. For example, in case of leaving the network, a node notifies all other participants about this event, and they adjust their local configuration files.

The core software component of the test-bench represents a universal control script that is loaded onto each node and can perform the functions of each of the described roles. The script contains a range of classes, and each cone is in charge of implementing a particular role of the node. When the network is initiated first time, the configuration file is read, and each node starts to function in accordance with the role assigned to it within this configuration. The new role of a node is enforced just after it is changed in the configuration file. This allows nodes to change roles in a fairly short time and seamlessly, i.e., without restarting the software of the node.

Table 7. Test-bench characteristics.

Name	Characteristics
Raspberry Pi model 3	CPU: Broadcom BCM2837 1.2ГГц 4 core ARM Cortex-A53; GPU: 2 core VideoCore IV; RAM: 1ГБ LPDDR2; Power: Micro USB socket 5.1В/2.5А (from powerbank) OS: Raspbian OS
Digi Zbee Series 2	Frequency: 2.4. . .2.4835 GHz Operating radius: 100 m Data transmission speed via radio channel: 250,000 bit/s Current consumption in receive mode: 50 mA Current consumption in power saving mode: <10 μA Count of channels: 16 Network addressing capabilities: PAN ID—16 bit, Address—16 bit Interfaces: UART, GPIO, PWM, ADC
Control software	Program language: Python 3.8 Specific libraries: Digi-XBee

Figure 8 schematically shows the used test-bench. The figure illustrates the presence of self-organization in the WSN (a case of adding a new node). The following condition for the redistribution of roles is also shown, namely the appearance of a node in the network that allows for optimizing the functions of the data keeper role. Also, the figure exposes an example of a new node appearing in the network, which leads to a redistribution of roles. In this case, such redistribution results in optimization of the function of the keeper role.

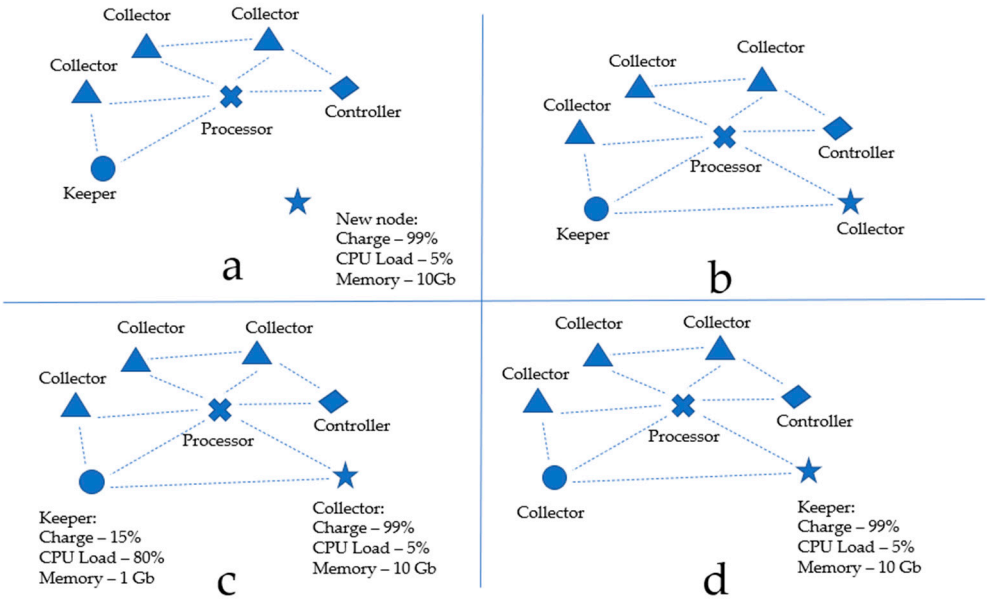

Figure 8. Illustration of a self-organizing decentralized WSN operation in the case of (**a**) new node appearing, (**b**) assigning a role to a new node, (**c**) initiating a redistribution of roles and (**d**) redistributing roles in the network.

In Figure 8, block 'a' illustrates the functioning of WSN with a certain set of roles for the nodes, where a new node with specific characteristics is added. Block 'b' exposes that a new node was successfully added to the WSN, the data transmission routes were rebuilt and the node started functioning with the default data collector role. Block 'c' illustrates a case where a node performing the data keeper role can no longer perform it since it has a low remaining battery charge and a low amount of remaining memory. In this case, the

node informs the network controller about this, and the process of role redistribution starts; the keeper role will then pass to another node (block 'd' in Figure 8).

The developed WSN model is primarily focused on solving issues of attack detection. Therefore, during a network operation, a log of events is recorded. The log collects events such as the initiation of reconfiguration of roles in the network and the deletion or addition of network nodes. Automated log analysis allows for the detection of attacks related to the exploitation of the self-organization and role-based functioning of the WSN. Log entries contain information about the time of an event, the type of event and its status. An example of a log recorded during testing is depicted in Figure 9.


```
{"timestamp": 1649786978.2143185, "event": "session_init", "state": 0}
{"timestamp": 1649786980.2143185, "event": "session_init", "state": -1}
{"timestamp": 1649786990.2143185, "event": "reconfiguration", "state": 0}
{"timestamp": 1649787000.2143185, "event": "session_init", "state": -1}
{"timestamp": 1649787100.2143185, "event": "session_init", "state": -1}
{"timestamp": 1649787203.2143185, "event": "reconfiguration", "state": 0}
{"timestamp": 1649788654.2143185, "event": "session_init", "state": 0}
```


Figure 9. A fragment of WSN event logs.

A validation scheme was developed to validate the proposed data collection and attack detection algorithms and their application technique. This scheme is presented in Figure 10.

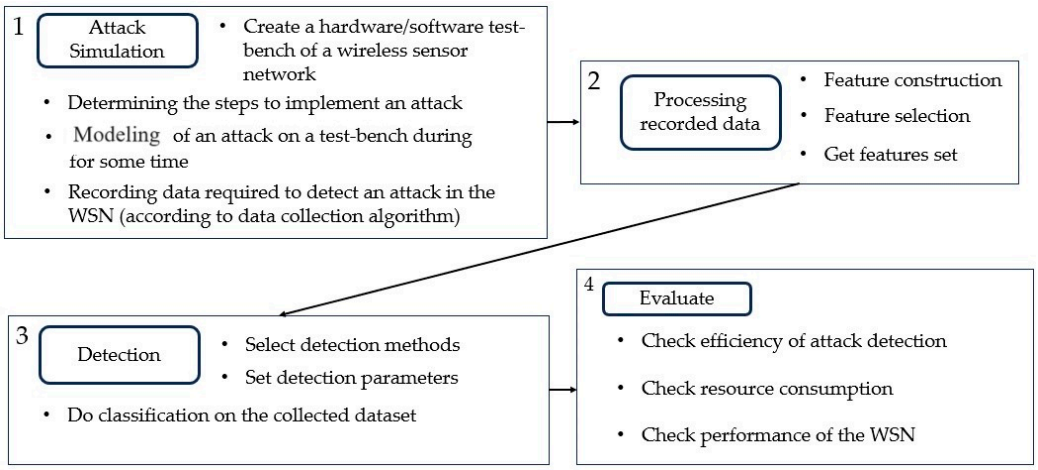

Figure 10. Validation scheme of the proposed solutions.

The first step of the validation scheme is to simulate the attack on a hardware/software test-bench. This step includes the creation of a prototype, declaring specific attack steps to be simulated, modeling the attack and recording the required data from the WSN by using the proposed data collection algorithm. For example, when implementing a flood attack on the test-bench, the specific steps of this attack were modeled, and they are exposed in Figure 9. Namely, the attacker, exploiting the property of self-organization, connects a malicious node to the network. To complete this activity, initially, the attacker learns the identifier of the network (PAN ID). Next, the attacker node initiates the process of redistribution of node roles in WSN. It waits for a response from other nodes in the WSN with a confirmation that they are ready for the redistribution. Just after that, the attacker terminates this initiation process. After the network returns to its initial state, the attacker repeats all these steps several times for a certain period (i.e., forming a long-duration repeated attack). In our experiments, the duration of running the test-bench was 60 min and the total duration of the attack was 20 min (namely, it was 4 attack runs of 5-min duration). Thus, the result of the first step is a recorded dataset, which is further used to train the classifiers.

The second step is the processing of the collected data. It includes extraction of the features required for the detection, constructing new features and generating a final dataset, which is then used to train detection models. The third step is the implementation of attack detection, i.e., the selection of detection methods and their hyper-parameters as well as training and classification. The fifth, and last, step is to evaluate the detection of quality as well as other metrics such as detection algorithm resource consumption. Specifically, common metrics, such as f-score, recall, precision and accuracy, are used to evaluate the detection quality. In addition, the detection algorithms are tested for false positives and missed attack incidents, and the time between the attack start and the successful detection is measured. The following values are set as the required values: the attack detection quality index f-score ≥ 0.9, while the time from the start of an attack to its detection should be no more than 1 minute. The validation of the proposed solutions is performed on the hardware/software test-bench of a wireless sensor network according to the previously described scheme.

The software/hardware test-bench of a self-organizing decentralized WSN is used to confirm the correctness and prospects of the proposed models and algorithms for attack detection. Logging main events in the network, such as adding a node or reconfiguring roles, makes it possible to implement mechanisms for detecting attacks related to, for instance, Denial-of-Service or unauthorized increase in resource consumption. In case of a Denial-of-Service attack mounted through the ability to add and remove network nodes, the remaining nodes start to constantly update their configuration files. These events lead to increased resource consumption and energy consumption. Thus, through the analysis of the event log, it is possible to detect an inflated frequency of adding or deleting network nodes, which can be used as one of the important features of such attacks.

To test the proposed data collection algorithm and attack detection algorithm, a Denial-of-Service attack (flood attack) was simulated on the test-bench of a self-organizing decentralized WSN with role-based functioning. Schematically, the simulation is presented in Figure 11. In essence, the attack consists of connecting a malicious node to the network and unauthorized initiation of the mechanism for redistributing roles. It is assumed that the attacker performs the following steps to realize this attack: connecting a compromised node to the WSN; initiating a request for redistribution of the roles of the nodes; incorrect completion of the role's redistribution request; returning the network to its original state; repetition of all previous steps several times.

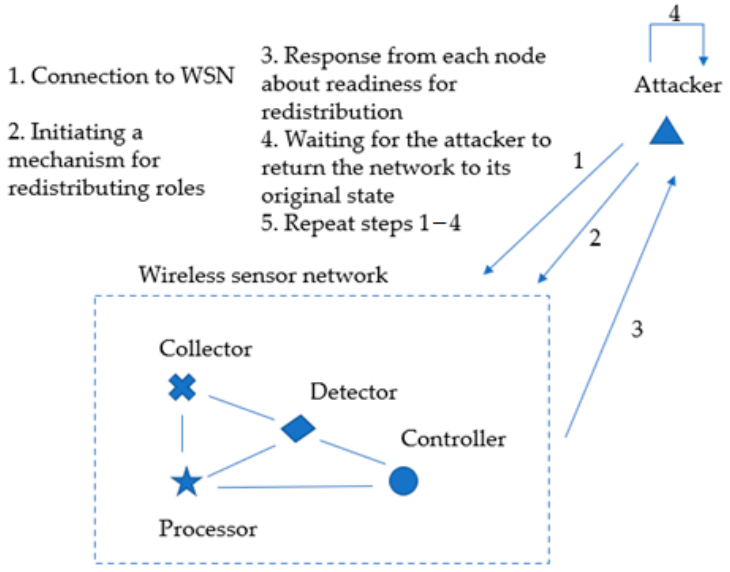

Figure 11. Illustration of attack impacts by simulating a flood attack.

As a result of this attack, all WSN nodes are constantly forced to spend additional computing resources and energy resources on service responses that are necessary for the normal process of role redistribution. However, the redistribution of roles is completed incorrectly, and the WSN returns to its original state, and so forth. Ultimately, the network's performance is disrupted, and the autonomous life of its nodes is significantly reduced.

In the process of modeling this attack, a dataset was collected. This dataset contains sensor readings from collectors transmitted over the network, as well as entries of the service log of the network controller, which includes requests for network redistribution and reconfiguration. The performed experiments on the test-bench confirmed that the collected data were sufficient to correctly detect the simulated attack. Some initial data that are required for the detection but are not explicitly in the logs are calculated by using statistical methods. For example, the value of the average frequency of role redistribution of nodes is calculated by using information from the service log of the network controller.

A few instances of detection modules for several variations of attacks were built. To produce these attack detection modules, machine learning methods realized in the scikit-learn library were used. The following classifiers were used: AdaBoost, Random Forest, Bayesian classifier (MultinomialNB), LogisticRegression, linear SVM classifier, decision trees and RidgeClassifier. The dataset constructed for the experiments contained data for the following types of attacks:

- Impacts on user and service data in the WSN (i.e., two variations, a common attack and one with the exploitation of the properties of self-organization and decentralization). These impacts involve distortion of the readings of WSN sensor values, as well as distortion of service data (sinkhole attack);
- Impacts on network devices (two variations, a common attack and one with the exploitation of the properties of self-organization and decentralization). These attacks include physical and other impacts on the nodes (hello-flood attack).

The results of the experiment are presented in Table 8. It discloses the name of the better particular ML method for detecting each type of impact and the values of the detection quality indicator (f-score was chosen as the most representative single indicator). In addition to the f-score, we measured, first, the network throughput, second, the increase in resource consumption with the algorithms embedded in contrast to the solution without the detection mechanism and, third, the delay before the attack detection (i.e., the time it takes for an attack to be detected after its launch). During the experiments on modeling and detection of the attacks, the metrics take the following values: average network throughput is 89%, the average delay before the attack detection is 1.2 s and the average increase in resource consumption is 18%. In all the experiments conducted, the increase in resource consumption of WSN nodes (i.e., CPU load in this case) after the implementation of data collection and attack detection algorithms did not exceed 20% of the initial one.

Table 8 discloses the outcome of experiments on attack detection in particular instances of attack variations. It demonstrates the ability of the proposed algorithms to detect not only common variations of the attacks but also those implemented by exploiting the properties of self-organization and decentralization. In order to optimize the resource consumption of the detector node, the case of combined use of the decision tree machine learning method for all four variations of attacks was also considered. The obtained high values of the detection quality metrics confirm the correctness of the proposed data collection and attack detection algorithms. In particular, for sinkhole and hello-flood, the experiments showed that the collected data were sufficient to build enough efficient attack detection algorithms to produce high-quality attack detection rates (Table 7). Considering a certain similarity of the input data for the detection of the different types of attacks considered in this article, this fact allows us to conclude that the developed algorithms for data collection and attack detection will work correctly for other attacks taken from the proposed attack model. However, for other types of attacks, these algorithms may require some adjustments to certain parameters, such as the types of data to be collected, as well as adjustments to the machine learning methods and their hyper-parameters.

Table 8. Attack detection quality indicators.

Attack Type	Variation	Machine Learning Method	F1 Score	Delay before Attack Detection, s	Average Throughput, %	Average Nodes CPU Load (%) (w/o Detection Algorisms/with Them)
Sinkhole	Common attack variation	SVM	0.99	1.2	86	34.5/40.3
	With the exploitation of the properties of self-organization and decentralization	SVM	0.98	1.0	88	33.1/38.4
Hello-flood	Common attack variation	Decision tree	0.99	1.3	87	54.3/65.7
	With the exploitation of the properties of self-organization and decentralization	Decision tree	0.99	1.1	89	55.6/63.9
Integral for the listed attacks types		Decision tree	0.98	1.2	89	59.0/70.8

Let us compare the obtained results with the detection results of similar attacks from alternative works. Generally, Table 1 in Section 2 reveals the quality results of detecting attacks in wireless sensor networks. For example, in [16], Chinnasamy et al. present a method to detect DDoS attacks with high accuracy, including flood attacks. However, other than the accuracy metric, the authors do not expose other metrics for evaluating the detection performance. In [12,14], methods for detecting various attacks in WSNs (including those similar to the sinkhole attack) as well as anomalies are discussed. The shown accuracy results for attack and anomaly detection are also quite high, i.e., the accuracy ranges from 0.83 to 1.0 for different types of attacks and 0.899 for anomaly detection. But note that the attacks considered by these authors are common variations of attacks on WSNs, i.e., they do not take into account the mechanism of decentralization and self-organization in the process of attack implementation. Note that, in the example of the flood attack discussed in this section, the attacker uses the mechanism of network decentralization, namely role allocation. Therefore, we can conclude that the detection algorithms proposed in this article work with high accuracy, both in the case of a new variation of attacks (exploiting the properties of decentralization and role allocation) and in the detection of the common variation of the attack.

The differences in the data collection algorithm include a decentralized method of data collection, being adapted during the operation, depending on, first, the current set of network nodes, second, the distribution of roles between them, third, the volume of available storage resources on the nodes and, finally, the size of the collected data flows.

The novelty of the attack detection algorithm consists of the constructed sets of features (1) obtained by analysis of the data collected in the WSN, network specifications and configurations, and (2) used to detect attacks that exploit the properties of decentralization and self-organization networks through using machine learning and statistical methods.

The technique for the model and algorithm application in self-organizing decentralized WSNs describes the main stages of the data collection and attack detection processes. In essence, it specifies a proper way for realization of the developed models and algorithms into a real physical implementation of a WSN. Its differences include experimentally substantiated stages for specifying and adjusting the parameters of data collection algorithms and detection algorithms, including the selection of hyper-parameters of the used learning methods.

Note that the presented algorithms, models and technique could be applied to research areas other than the considered one, for instance, combined Ethernet/Wi-Fi networks and ad hoc smart device networks. For example, a sinkhole attack, which redirects traffic

through an attacker node, and a Man-in-the-Middle attack being close to sinkhole, are relevant to Ethernet/Wi-Fi networks. When this attack is mounted, data confidentiality and integrity are compromised, and, typically, it is important the network is operated in a critical infrastructure.

To detect Man-in-the-Middle attacks for a specific system, it may be necessary to adjust the types of input data for detection due to the different methods and the differing nature of the communication. Nevertheless, the results of this article could be relevant to a broader class of networks but require further research to validate the applicability and effectiveness of the proposed modeling framework for the broader class of networks and cyber–physical systems.

It is worth noting that the developed algorithms and technique have certain limitations, primarily concerning resource constraints. These limitations include constraints on the duration and volume of data to be stored on the nodes and needed for the attack detection. In essence, WSN nodes represent devices with limited hardware and software capabilities as well as energy resources, often not suitable for installing heavy databases, and should not be used to store large amounts of data for a long time. Therefore, one of the most important aspects is to use lightweight data storage only, provided that severe restrictions are placed on the volume of data and the duration of storage. In the experimental example, the data were stored in JSON format, and the duration of data stored simultaneously was 1 h, with 10 GB of the occupied volume (i.e., data older than one hour of the system operation and data more than 10 GB were discarded).

Another limitation of the proposed algorithms is that they should not consume more resources than allowed. That is, the amount of resource consumption of the system with the algorithms implemented should not exceed the specified limit (in fact, in the regarded case, the increase in the consumption of computational resources does not exceed 20%). In addition to the limitations, the following requirements were imposed on the developed solutions: maximization of the number of detected attack types and preservation of the WSN operability (i.e., an ability to continue to perform the main functions of the WSN still correctly after the embedded data collection and attack detection algorithms start working).

It is also worth noting that the solutions described in the article, including the attack model, can be used at the design stage of such self-organized decentralized WSNs in order to facilitate improving its security in the process of its further design and development process. For example, to protect against DoS attacks, such as hello-flood, it is possible to limit the allowed number of new nodes to be connected to the network in a certain time interval.

Alternatively, to protect against malicious code injection attacks, it is reasonable to add checking of incoming data frames, including checking for special characters and possible other incorrect frame content. As an additional protection measure, it is possible to consider some reliable protection of nodes from software breaking, for example, the installation of a solid authentication component and other tamper-resistant means. This measure will prevent an intruder from copying the WSN settings (i.e., network identifier, list of roles, business algorithm running on the nodes) on malicious nodes, and its implantation into the network will be complicated.

Let us consider another example. To protect it against an attack when a malicious node illegally obtains some specific role in WSN, it is possible to limit the number of role reassignments in the network per unit of time. Another alternative would be to implement a mechanism to check a node for compliance with the requested role (i.e., availability of a large amount of storage for nodes with the role of the data keeper or sufficient computing resources for nodes with the role of data processor).

Based on this analysis, the following practical recommendations that should be considered when designing such a WSN could be proposed:

- Conducting analysis of relevant types of attacks and constantly updating them, considering the specificity of the particular application field;

- In case of detection of new types of attacks, one should perform a detailed analysis of available data (especially data on self-organization and decentralization to be extracted from logs as well as data from sensors) for their suitability for detection of new attacks;
- To complicate attacks exploiting decentralization of WSN, an adjustment in the network security policy to reduce the allowed opportunities for a priori legitimate network rebuilding should be performed;
- To set the minimum required and secure settings of WSN operation, for instance, a number of nodes added per unit of time, number of requests for role redistribution per unit of time, using strict specifications of requirements for nodes for each role, etc. In particular, fulfillment of these restrictions will reduce the risk of DoS attacks, as well as data replay and wormhole attacks;
- To provide reliable authentication on the WSN nodes in order to prevent unauthorized access to them and theft of parameters and settings of the WSN nodes;
- To compare and select the most profitable machine learning methods as attack detection means that allow for high-precision detection.

9. Conclusions

This study examines self-organizing decentralized WSNs with role-based functioning of nodes. The main contribution is made by a model of attacks on this kind of WSN, algorithms for data collection and attack detection, as well as a technique for their application. The WSN model is presented by state and transition graphs and a set of functional roles of the nodes. The developed model allows us to formalize the process of WSN functioning and its inherent properties of self-organization and decentralization.

A distinctive feature of the proposed attack model is that it uses properties of self-organization and decentralization of the network, as well as role-based functioning. More specifically, it allows us to take into account the influence of these properties on the attacks and methods of their detection. The data collection algorithm specifies the types of data needed. The algorithm for detecting attacks in self-organizing wireless sensor networks consists of choosing ML methods and their hyper-parameters to detect relevant attacks on WSNs with high detection quality, taking into account the given constraints, namely the constraints on the duration of data storage for detection (no more than 1 h of network operation), resource consumption by the implemented algorithms (consumption of no more than 20% of computational resources) and under the conditions of preserving the WSN operability.

In addition, this article describes the particular kinds of features needed for the detection of the attacks. The proposed technique joins the developed models and algorithms as well as outlines the methods of their combined application in practice.

In the future, we plan to expand the experiments on attack detection by simulating other various kinds of attacks targeted at self-organizing decentralized WSNs with the help of the used software/hardware test-bench. It is also planned to test this complex approach to attack detection in the field of autonomous vehicles [36]. Vehicle swarms can be carriers of WSNs and could be used to extend the range of ground-based WSNs, and they can also perform the function of measuring environmental parameters. Since such swarms are also susceptible to attacks, further research in the field of their security becomes especially relevant.

Author Contributions: Conceptualization, V.D. and A.M.; methodology, A.M. and V.D.; investigation, V.D. and A.M.; software A.M. and V.D.; writing—review and editing, A.M. and V.D.; Project administration, V.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author (after the approval of the organization).

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

References

1. Ortiz, A.M.; Olivares, T.; Orozco-Barbosa, L.; Perez-Juana, M. Measurements with Different Role-based Wireless Sensor Network Organizations. In Proceedings of the 3rd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks, Vancouver, BC, Canada, 31 October 2008; pp. 9–16.
2. Durrani Tariq, S.; Wang Wei Forbes Sheila, M. A Survey on the Role of Wireless Sensor Networks and IoT in Disaster Management. *Geol. Disaster Monit. Based Sens. Netw.* **2019**, 57–66. [CrossRef]
3. Zahraa, T.A.; Salah, A.A. A survey of disaster management and SAR operations using sensors and supporting techniques. *Int. J. Disaster Risk Reduct.* **2022**, *82*, 103295. [CrossRef]
4. Erdelj, M.; Krol, M.; Natalizio, E. Wireless Sensor Networks and Multi-UAV systems for natural disaster management. *Comput. Netw.* **2017**, *124*, 72–86. [CrossRef]
5. Kochhal, M.; Schwiebert, L.; Gupta, S. *Role-Based Middleware for Sensor Networks*; Wayne State University: Detroit, Michigan, 2004.
6. Hong, M.; Jinman, J.; Seoyeon, K.; Bongjae, K.; Junyoung, H. Role-based automatic programming framework for interworking a drone and wireless sensor networks. In Proceedings of the 33rd Annual ACM Symposium on Applied Computing—SAC '18, Pau, France, 9–13 April 2018; pp. 1853–1856. [CrossRef]
7. Sudip, M.; Ankur, V. Reputation-based role assignment for role-based access control in wireless sensor networks. *Comput. Commun.* **2011**, *34*, 281–294.
8. Ruairi, R.; Keane, M. The Dynamic Regions Theory: Role Based Partitioning for Sensor Network Optimization. In Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems, London, UK, 29 May–2 June 2023.
9. Kumar, R.; Wolenetz, M.; Agarwalla, B.; Shin, J.S.; Hutto, P.; Paul, A.; Ramachandran, U. DFuse: A Framework for Distributed Data Fusion. In Proceedings of the First International Conference on Embedded Networked Sensor Systems, Los Angeles, CA, USA, 5–7 November 2003.
10. Dasgupta, K.; Kukreja, M.; Kalpakis, K. Topology-aware placement and role assignment for energy-efficient information gathering in sensor networks. In Proceedings of the eight IEEE International Symposium on Computers and Communications, Kemer-Antalya, Turkey, June 30–3 July 2003.
11. Ortiz, A.; Olivares, T.; Orozco-Barbosa, L. A heterogeneous role-based sensor network. In Proceedings of the 2nd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks, Chania Crete Island, Greece, 22 October 2007. [CrossRef]
12. Silva, A.R.; Martins, H.T.; Rocha, P.S.; Loureiro, A.F.; Ruiz, B.; Wong, H.C. Decentralized intrusion detection in wireless sensor networks. In Proceedings of the 1st ACM International Workshop on Quality of Service & Security in Wireless and Mobile Networks (Q2SWinet '05), Montreal, QC, Canada, 13 October 2005; pp. 16–23.
13. Huang, K.; Yuen, K. Online dual-rate decentralized structural identification for wireless sensor networks. *Struct. Control. Health Monit.* **2019**, *26*, e2453. [CrossRef]
14. Safaei, M.; Ismail, A.S.; Chizari, H.; Driss, M.; Boulila, W.; Asadi, S.; Safaei, M. Standalone noise and anomaly detection in wireless sensor networks: A novel time-series and adaptive Bayesian-network-based approach. *Softw. Pract. Exp.* **2020**, *50*, 428–446. [CrossRef]
15. Kohno, E.; Ohta, T.; Kakuda, Y. Secure decentralized data transfer against node capture attacks for wireless sensor networks. *Int. Symp. Auton. Decentralized Syst.* **2009**, 1–6. [CrossRef]
16. Chinnasamy, P.; Devika, S.; Balaji, V.; Dhanasekaran, S.; Jebamani BJ, A.; Kiran, A. BDDoS: Blocking Distributed Denial of Service Flooding Attacks With Dynamic Path Detectors. In Proceedings of the 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 23–25 January 2023; pp. 1–5. [CrossRef]
17. Chinnasamy, P.; Kumaresan, N.; Selvaraj, R.; Dhanasekaran, S.; Ramprathap, K.; Boddu, S. An Efficient Phishing Attack Detection using Machine Learning Algorithms. In Proceedings of the 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC), Bhubaneswar, India, 19–20 November 2022; pp. 1–6. [CrossRef]
18. Yingxu, L.; Liyao, T.; Jing, L.; Yipeng, W.; Tong, T.; Zijian, Z.; Hua, Q. Identifying malicious nodes in wireless sensor networks based on correlation detection. *Comput. Secur.* **2022**, *113*, 102540. [CrossRef]
19. Bhardwaj, A.; Tyagi, R.; Sharma, N.; Khare, A.; Punia, M.S.; Garg, V.K. Network intrusion detection in software defined networking with self-organized constraint-based intelligent learning framework. *Meas. Sens.* **2022**, *24*, 100580. [CrossRef]
20. Kumar, V.N.; Srisuma, V.; Mubeen, S.; Mahwish, A.; Afrin, N.; Jagannadham DB, V.; Narasimharao, J. Anomaly-Based Hierarchical Intrusion Detection for Black Hole Attack Detection and Prevention in WSN. In Proceedings of the Fourth International Conference on Computer and Communication Technologies. Lecture Notes in Networks and Systems, Orlando, FL, USA, 8–10 May 2023; Springer: Berlin/Heidelberg, Germany, 2023; Volume 606. [CrossRef]
21. Jane Nithya, K.; Shyamala, K. A Systematic Review on Various Attack Detection Methods for Wireless Sensor Networks. In Proceedings of the International Conference on Innovative Computing and Communications. Advances in Intelligent Systems and Computing, Delhi, India, February 2022; Volume 1394. [CrossRef]
22. Sahu, M.; Sethi, N.; Das, S.K. A Survey on Detection of Malicious Nodes in Wireless Sensor Networks. In Proceedings of the 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 28–30 April 2022; pp. 710–715. [CrossRef]
23. Grover, J.; Sharma, S. Security Issues in Wireless Sensor Network—A Review. In Proceedings of the 5th International Conference on Reliability. Infocom Technologies and Optimization (ICRITO), Noida, India, 7–9 September 2016; pp. 397–404.

24. Rehman, A.; Rehman, S.U.; Raheem, H. Sinkhole Attacks in Wireless Sensor Networks: A Survey. *Wirel. Pers. Commun.* **2019**, *106*, 2291–2313. [CrossRef]
25. Ahutu, O.R.; El-Ocla, H. Centralized Routing Protocol for Detecting Wormhole Attacks in Wireless Sensor Networks. *IEEE Access* **2020**, *8*, 63270–63282. [CrossRef]
26. Ghugar, U.; Pradhan, J. Survey of wormhole attack in wireless sensor networks. *Comput. Sci. Inf. Technol.* **2021**, *2*, 33–42. [CrossRef]
27. Shakhov, V.V.; Yurgenson, A.N.; Sokolova, O.D. Modeling and simulation of Black Hole attack on wireless sensor networks. *Softw. Prod. Syst.* **2017**, *30*, 34–39.
28. Alahari, H.P.; Yelavarthi, S.B. Performance Analysis of Denial of Service DoS and Distributed DoS Attack of Application and Network Layer of IoT. In Proceedings of the 2019 Third International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 10–11 January 2019; pp. 72–81. [CrossRef]
29. Nwokoye, C.H.; Madhusudanan, V. Epidemic Models of Malicious-Code Propagation and Control in Wireless Sensor Networks: An Indepth Review. *Wirel. Pers. Commun.* **2022**, *125*, 1827–1856. [CrossRef]
30. Radhika, S.; Anitha, K.; Kavitha, C.; Lai, W.-C.; Srividhya, S.R. Detection of Hello Flood Attacks Using Fuzzy-Based Energy-Efficient Clustering Algorithm for Wireless Sensor Networks. *Electronics* **2023**, *12*, 123. [CrossRef]
31. Islam, M.N.U.; Fahmin, A.; Hossain, M.S.; Atiquzzaman, M. Denial-of-Service Attacks on Wireless Sensor Network and Defense Techniques. *Wirel. Pers. Commun.* **2021**, *116*, 1993–2021. [CrossRef]
32. Amirreza, Z.; Behrouz, S.; Wolfgang, B. Security analysis and fault detection against stealthy replay attacks. *Int. J. Control.* **2022**, *95*, 1562–1575. [CrossRef]
33. Adil, M.; Almaiah, M.A.; Omar Alsayed, A.; Almomani, O. An Anonymous Channel Categorization Scheme of Edge Nodes to Detect Jamming Attacks in Wireless Sensor Networks. *Sensors* **2020**, *20*, 2311. [CrossRef]
34. Jeyaselvi, M.; Sathya, M.; Suchitra, S.; Jafar Ali Ibrahim, S.; Kalyan Chakravarthy, N.S. SVM-Based Cloning and Jamming Attack Detection in IoT Sensor Networks. *Adv. Inf. Commun. Technol. Comput. Lect. Notes Netw. Syst.* **2022**, *392*, 461–471. [CrossRef]
35. Meleshko, A.; Desnitsky, V.; Kotenko, I. Approach to Anomaly Detection in Self-Organized Decentralized Wireless Sensor Network for Air Pollution Monitoring. *MATEC Web. Conf.* **2021**, *346*, 1–8. [CrossRef]
36. Chevalier, Y.; Fenzl, F.; Kolomeets, M.; Rieke, R.; Chechulin, A.; Krauss, C. Cyberattack detection in vehicles using characteristic functions, artificial neural networks and visual analysis. *Inform. Autom.* **2021**, *20*, 845–868. [CrossRef]