electronics

MDPI

Article

A Secure Protocol against Selfish and Pollution Attacker Misbehavior in Clustered WSNs

Hana Rhim [1,*], Damien Sauveron [2], Ryma Abassi [1], Karim Tamine [2] and Sihem Guemara [1]

[1] Digital Security Lab, Sup'Com, University of Carthage, Tunis 2083, Tunisia; ryma.abassi@supcom.tn (R.A.); sihem.guemara@supcom.tn (S.G.)
[2] MathIS, XLIM, University of Limoges, 87032 Limoges, France; damien.sauveron@unilim.fr (D.S.); karim.tamine@unilim.fr (K.T.)
* Correspondence: hana.rhim@supcom.tn; Tel.: +216-54-702-104

Abstract: Wireless sensor networks (*WSNs*) have been widely used for applications in numerous fields. One of the main challenges is the limited energy resources when designing secure routing in such networks. Hierarchical organization of nodes in the network can make efficient use of their resources. In this case, a subset of nodes, the cluster heads (*CH*s), is entrusted with transmitting messages from cluster nodes to the base station (*BS*). However, the existence of selfish or pollution attacker nodes in the network causes data transmission failure and damages the network availability and integrity. Mainly, when critical nodes like *CH* nodes misbehave by refusing to forward data to the *BS*, by modifying data in transit or by injecting polluted data, the whole network becomes defective. This paper presents a secure protocol against selfish and pollution attacker misbehavior in clustered *WSNs*, known as (*SSP*). It aims to thwart both selfish and pollution attacker misbehaviors, the former being a form of a Denial of Service (*DoS*) attack. In addition, it maintains a level of confidentiality against eavesdroppers. Based on a random linear network coding (*NC*) technique, the protocol uses pre-loaded matrices within sensor nodes to conceive a larger number of new packets from a set of initial data packets, thus creating data redundancy. Then, it transmits them through separate paths to the *BS*. Furthermore, it detects misbehaving nodes among *CH*s and executes a punishment mechanism using a control counter. The security analysis and simulation results demonstrate that the proposed solution is not only capable of preventing and detecting *DoS* attacks as well as pollution attacks, but can also maintain scalable and stable routing for large networks. The protocol means 100% of messages are successfully recovered and received at the *BS* when the percentage of lost packets is around 20%. Moreover, when the number of misbehaving nodes executing pollution attacks reaches a certain threshold, *SSP* scores a reception rate of correctly reconstructed messages equal to 100%. If the *SSP* protocol is not applied, the rate of reception of correctly reconstructed messages is reduced by 90% at the same case.

Keywords: secure routing; network coding; wireless sensor network; data availability; selfish behavior; *DoS* attack; data redundancy; integrity; pollution attack; confidentiality; eavesdropping attack; energy efficiency; security; wiretapper; hierarchical routing; cluster head; scalability

Citation: Rhim, H.; Sauveron, D.; Abassi, R.; Tamine, K.; Guemara, S. A Secure Protocol against Selfish and Pollution Attacker Misbehavior in Clustered WSNs. *Electronics* **2021**, *10*, 1244. https://doi.org/10.3390/electronics10111244

Academic Editor: Antonio Pescapè

Received: 19 December 2020
Accepted: 10 May 2021
Published: 24 May 2021

1. Introduction

Wireless sensor networks (*WSNs*) are receiving much attention from both research and industrial fields. The main attractive aspects are their low cost, and effective and easy-to-implement sensors. They are re-configurable to many types of applications to solve real-world problems. However, these small sensors are resource-constrained and battery-limited. In addition, they can be deployed in open and remote environments on large scales that are possibly exposed to several risks. Consequently, they are vulnerable to numerous security attacks. Counteracting these attacks and implementing security techniques can involve a great energy cost to these networks. It can damage the balance of

energy consumption in the network, decreasing the nodes' connectivity and the network survival time.

One way to preserve some battery power during communications involves the choice of network architecture and of the protocol used to route data from a sensor to the *BS*. A number of methods designed to handle security issues in *WSNs* exploit network hierarchization and cluster-based routing [1]. Clustering a sensor network typically leads to possibilities for scaling, better control of nodes and energy efficiency gains provided by partitioning [2,3]. Other ways to preserve battery power are sleep scheduling and energy harvesting. The former aims at minimizing the number of sensors to activate to cover a desired portion of the region of interest preserving the connectivity among sensors [4]. The latter aims to resolve the issue that the nodes are often unreachable after deployment and introduces the concept of renewable energy that can be harvested from the surrounding environment [5]. As mentioned before, *WSNs* are vulnerable to several attacks, which can be classified into two types:inside attacks, performed by authorized nodes and outside attacks, performed by unauthorized nodes.

Previous studies have shown that inside attacks are far more difficult to control than outside ones [6,7]. Selfishness and pollution attacks are two forms of critical inside attacks, where nodes can easily launch a type of Denial of Service (*DoS*) in the first, or pollute data in transit to interrupt the normal functioning of the network, in the second. More specifically, in the former attack, selfish nodes refuse to relay packets and use network resources for their own benefit. This operation is one variety of a *DoS* attack which can be performed in different ways [8] and has multiple forms [9]. In fact, a *DoS* attack can also be achieved by congesting the network or draining the energy from its components [10]. This attack is considered one of the most frequent attacks in *WSNs* [11,12]. In the latter attack, pollution attacker nodes modify native data packets or inject fake ones into the network. A pollution attack is potentially more damaging when applied in protocols for *WSNs* based on network coding (*NC*) since it may have devastating impact on the data routing [13]. Figure 1 illustrates the aforementioned considered attacks.

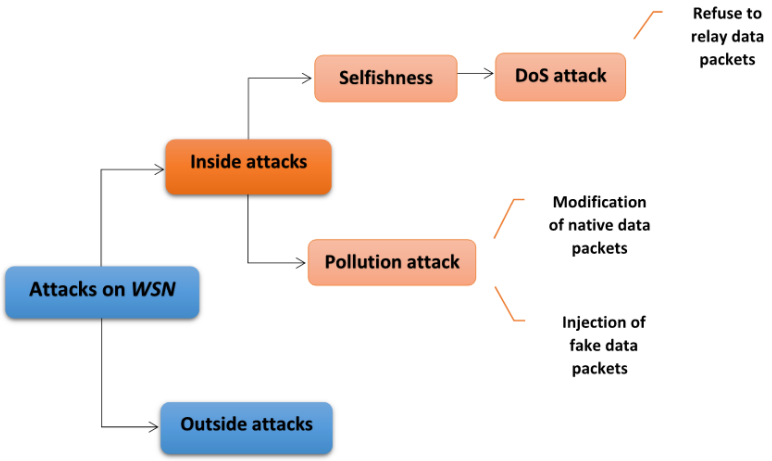

Figure 1. The inside attacks performed on the *WSN*.

In a clustered network, if the misbehaving nodes that invoke a selfishness or a pollution attack are the *CH* nodes, then the network will be more disordered and the consequences more serious. The *CH*s have a critical role in forwarding the cluster data to the *BS* [14]. *DoS* and pollution attacks are challenging issues that aim to disable the service that a *WSN* is supposed to provide, and create routing failure.

This paper proposes a secure protocol for *WSN*s that defends against the misbehavior of selfish and pollution attacker nodes in the network. It extends our previous work [15], which ensured confidentiality of exchanges in clustered *WSNs* using an optimized (*NC*) technique. This extension provides a solution for preventing an attack on network avail-

ability as well as integrity. It also proposes attack detection and a punishment mechanism for malicious nodes causing the *DoS* attack or the pollution attack. The solution takes advantage of the matrix format of keys specific to each sensor node used for confidential communication in [15] and implements a new scheme of *NC* by adding data redundancy. It converts a file/message into a set of p pieces in such a way that the file/message can be reconstructed from any predefined subsets of t out of p pieces. Consequently, our solution ensures data availability and integrity with considering the confidentiality requirement already existing.

Our proposition is a secure protocol against selfish and pollution attacker misbehavior in clustered *WSNs*, named *SSP*. It aims to shield against a coalition of misbehaving *CH* nodes that initiate a *DoS* attack by dropping received packets or a pollution attack by altering them. The main contributions of this paper are as follows:

1. Preventing a *DoS* attack by a group of *CH* nodes and ensuring a reliable transfer of data between source and *BS*.
2. Ensuring data integrity and error correction despite the existence of a pollution attack performed by a group of *CH* nodes.
3. Locating the misbehaving nodes among *CH*s and implementing a punishment mechanism against them to enhance the network availability and integrity.
4. Providing simulation results that prove the effectiveness of our protocol compared to a defenseless protocol, regarding the percentage of correct messages received and also considering the case when the number of misbehaving nodes is varying.
5. Proving the scalability of the protocol.

The present paper is structured as follows. Section 2 gives feedback on the literature dealing with existing secure protocols for *DoS* and pollution attacks prevention/detection. Section 3 presents the assumptions about the system and the threat model used in the paper. The design steps for the protocol and a theoretical discussion are given in Section 4. The simulation set-up and the results are presented in Section 5. Section 6 summarizes the paper and provides conclusions.

2. Related Work

In this section, we present feedback on the literature on secure protocols dealing with availability and integrity requirements in *WSNs*.

Many studies have been conducted to propose solutions that provide data availability to routing protocols, whether by preventing or detecting *DoS* attacks in *WSNs*.

In Ref. [16], the authors proposed a protection against *DoS* attacks in *WSNs*. Their solution is based upon two phases. The first phase partitions the network into clusters via the Hybrid Energy-Efficient Distributed clustering (*HEED*) protocol. The second sets up a Key Distribution Server (*KDS*), which supplies each node with session keys and unique *ID*s. Then, it executes a mutual authentication process between server and *CH*s and then between *CH*s and cluster members using a hash function. When a *CH* detects a malicious node after an unsuccessful authentication process, it requests the *KDS* to delete its secret key and requests all the other *CH*s to block it from inter-cluster communication. Consequently, this node becomes keyless and all the services related to it are arrested. After a certain period of time, the *KDS* calculates new session keys for each *CH* to communicate with the server and with the cluster members. Although the proposed mechanism can prolong the network lifetime and reduce the overhead using HEED clustering protocol, the cryptography-based mechanism used for defending *DoS* attacks is computationally expensive and also energy consuming.

Mansouri et al. designed a clustering-based approach to address *DoS* attacks in *WSNs* [17]. Their approach is conducted in two phases. The first phase elects a control node (*Cnode*) with a recursive low-energy adaptive clustering hierarchy (*LEACH*) processing algorithm, for each cluster. The second phase permits detecting and blocking of compromised nodes by the *Cnode*. When a node sends a number of messages that are greater than a threshold, the *Cnode* identifies that node as a compromised node. All messages

sent by the compromised node will be rejected and ignored by the neighboring nodes. The proposed protocol tends to provide significant results in term of time detection and throughput. However, it only deals with a specific type of *DoS* attack, which is excessive data transmission to drain the node's energy. In addition, it does not consider the case when a *CH* node executes the attack.

In Ref. [18], the authors adopted recursive clustering based on the *LEACH* algorithm. However, they enhanced it by using a novel algorithm named Fast and Flexible Unsupervised Clustering Algorithm (*FFUCA*). In this approach, the recursive clustering is used to identify the *CH* node and control nodes (*Cnode*) in each cluster, taking into account the node's location and energy consumed. The authors compared the relevance of these two algorithms (*FFUCA* and *LEACH*). *FFUCA* generates an optimal solution for minimizing distances between *Cnode* and sensor nodes when deploying the network compared to *LEACH*. It also achieves better *DoS* detection for false positive and false negative rates. However, this approach still does not deal with the case when the *CH*s are compromised. Sujit et al. proposed a new approach for detection of selfish node's behavior in *WSN* [19]. They used average and maximum values of re-transmissions in several nodes aimed to detect selfish nodes in the network. First, the protocol sets the shortest routes using Dijkstra's algorithm. Then, it decides on a node type, whether it is a partially selfish node, a fully selfish node or a non-selfish node, using a threshold value and the calculated maximum value of re-transmissions. This system detects but does not prevent selfish behavior when routing data in the network.

Virmani et al. [20] introduced an exponential trust-based mechanism to identify malicious nodes in a clustered network. Heads of clusters are selected based on their higher energy level. When a source node in a cluster sends a packet to a particular node, its streak counter stores the count of consecutive packets dropped. The mechanism calculates a trust factor (*TF*) formula, depending on the counter value, which falls exponentially with the increase in the number of packets dropped by the particular node. When *TF* goes below a certain threshold value, the node is stated as an adversary. In this case, the cluster head sends a request to the source node to re-transmit the packet. The proposition tends to detect the black hole attack, a type of *DoS* attack, where packets are consecutively dropped. However, it does not ensure protection from the attack and uses re-transmission of dropped packets, a process which has an extra energy cost.

Kalkha et al. [21] proposed an approach for preventing black hole attack in *WSN* using the Hidden Markov Model (HMM) to model the succession of choices of shortest path made by a source node to reach a destination node. Their approach uses a Viterbi algorithm to determine the path with the greatest probability of being malicious. Next, it identifies the likeliest fake nodes in the selected malicious path and sets a new routing algorithm that avoids these malicious nodes. The approach helps to avoid the selection of malicious path and node for routing data to the destination. However, it is dependent on a probabilistic, not a deterministic method. Moreover, it uses flat routing where each source node performs a multiple paths selection and re-selection after the maliciousness analysis, which will exhaust its resources quickly.

The network coding (*NC*) methodology has been used in many research papers to ensure security against various attacks [22]. Specifically, this methodology was introduced as an alternative to conventional networking, in which intermediate nodes merely forwarded incoming packets without alteration. In fact, it performs computations on received data prior to forwarding the data to the next hop. The *NC* technique exists in a range of types that are further clarified and detailed in [23]. We have proposed, in a previous study, a secure network coding-enabled approach for a confidential cluster-based routing in wireless sensor networks [15], called *SNCR*. Without the need to use expensive traditional cryptographic-based methods, it offers an optimized version of *NC* methodology to overcome eavesdropping attacks on transmitted data. It exploits pre-loaded hidden encoding vectors and transmits only a single digit instead of transferring all the coding coefficients with the coded vector. Therefore, it minimizes the overheads compared to

conventional network coding systems. *SNCR* has been proved to guarantee confidential data transmission whether the adversary attack is internal or external to the network. However, it does data coding at two hierarchical levels of the network, source nodes and *CH*s, which reduce the overall power of the network.

The specific communication mode of *NC*, in which intermediate nodes are allowed to deliberately change the received packets, creates opportunities for malicious nodes to disrupt correct data routing. When a packet (possibly encoded) is polluted, the decoding of the set of encoded packets associated with it will not generate the correct original message data at the destination node. Thus, a pollution attack in an environment where network coding is used is more likely, and preventing it becomes more important [24].

In this context, several solutions have been proposed to thwart pollution attacks in *NC*-enabled networks. Adeli et al. proposed in [25] a secure linear network coding scheme built upon a cryptographic hash function. This latter is used to introduce different random noisy terms within the information symbols. However, their scheme imposed a restriction on the linear network code design. A new refined scheme was proposed by Kim and Young-Sik in [26] to remove this restriction but this type of scheme still needs computation of the hash values of each packet, thereby increasing the transmission delay and the required operations' complexity.

Yu et al. proposed an efficient homomorphic signature scheme based on the *RSA* signature [27]. In their scheme, the source signs its message using its private key. Intermediate nodes verify the received messages using the source's public key. The authors used a novel homomorphic function where the encoded message's signature is composed from those of the input messages (used to create the encoded message). Their scheme can accomplish source authentication and data verification. However, it increases transmission overhead, since the size of an *RSA* signature is typically very large. The authors of [28] proposed an identity-based digital signature scheme to detect pollution attacks in intra-session network coding. Their scheme does not involve a third-party query to certificate authority and does not have the key issue, and it does not take into consideration the energy consumption parameter in resource-constrained *WSNs*.

Liu Xiang et al. [29] presented a privacy-preserving signature scheme for linear network coding-based networks. They used a homomorphic signature and applied new signing and verification processes that enhance the computational efficiency. Their scheme optimizes the inherent security potential of random network coding and provides countermeasures against both eavesdropping attacks and pollution attacks. By encrypting a message before signing it, the intermediate nodes are able to capture and discard fake packets. Although the homomorphic signature *NC* scheme can solve the problem of data integrity and confidentiality, the signature generation and verification processes remain high-energy expenses, reducing the transmission efficiency.

Another approach, proposed in [30], used an identity-based linearly homomorphic signature scheme for *NC*-enabled wireless sensor networks to ensure data integrity and authenticity. In their scheme, the authors applied a signature generation and verification that are both independent of the size of the data packet to reduce the computation cost. They used the unique serial numbers of source nodes as an identity-based public key that is free of certificate management, allowing the destination to track the source of data.

In the previous studies on solutions, whether for *DoS* attacks or pollution attacks, we noted several disadvantages. In the solutions tackling *DoS* attacks, some ignored the case where the *CH*s are the malicious nodes in the network, despite their major role in controlling data routing, which can affect network availability. Others focused only on solutions for detecting *DoS* attacks and not for preventing them. Others approaches used cryptographic mechanisms with high computational costs. Solutions attempting to resist pollution attacks, using hash functions or homomorphic signatures, are computationally expensive and may lead to traffic overhead.

Our solution tackles both *DoS* and pollution attacks, along with eavesdropping attacks. In contrast to the previous work concerning network availability, the solution proposes a

prevention as well as a detection scheme for misbehaving *CH* nodes. It ensures original packets reach their destination even if a number of packets are dropped. In addition, it considers the case where the *CH* nodes are malicious. Moreover, our solution takes advantage of the *NC* technique used in [15] to guarantee data confidentiality and optimizes it by achieving data coding only at source nodes. Hence, it minimizes overheads and energy depletion in the network. Second, in relation to the previous work attempting to oppose pollution attacks, the solution does not involve an extra cryptographic mechanism, which drains sensor node energy, to ensure data integrity; it requires only extra computations at the level of the *BS*, which is supposed to be an unlimited-resource device.

3. Model Establishment

In this section, we start by defining the basics of the *NC* technique and the modifications brought to it and used in this paper for ensuring data availability and integrity and minimizing the overheads in the network. Then, we introduce the assumptions used for the network and the sensor nodes. We finish this section by presenting the threat model along with the targeted security objectives upon which our approach is built.

3.1. About the Network Coding Technique

We begin by introducing the basic principles of the standard *NC* and then we present the modified version of *NC* that uses data redundancy to protect against *DoS* and pollution attacks.

In *NC*-enabled networks, intermediate nodes are able to re-code and combine the received packets in order to generate new ones. More specifically, a fixed set of original data messages received at a relay node r from different source nodes is linearly combined in a hierarchical network to generate new coded messages. As a result that the combination is linear, the technique is called linear network coding (*LNC*) [31,32]. Let us note $x = (x_1, x_2, \ldots, x_L)$ the set of original data messages of size L. Once received at a relay node r, a resulting coded packet y is generated using an encoding vector $v = (v_1, v_2, \ldots, v_L)$. Eventually, it processes the coding operation as follows:

$$y = v_1 \times x_1 + v_2 \times x_2 + \ldots + v_L \times x_L \quad (1)$$

If two data packets meet in the network at a relay node, the node can derive a new linear combination using random encoding vectors. The standard *NC* technique thus allows data re-coding multiple times. The final destination, which is supposed to be the *BS*, needs to receive at least L coded packets $(y_1, y_2, \ldots, y_L)$ from different relay nodes to be able to retrieve the original data messages $(x_1, x_2, \ldots, x_L)$ using the same encoding vectors $v = (v_1, v_2, \ldots, v_L)$. Figure 2 shows an example of *NC*, where nodes *S* and *D* correspond to the source node and the destination node, respectively. It illustrates the way the *NC* technique operates with a set of original data messages of size 2.

There are many existing *NC* techniques in the literature. Ref. [23] depicts the evolution of *NC* theory and its families.

In order to ensure availability and integrity and counteract the misbehavior of the adversary, we incorporate new features into the *NC* technique. In fact, the *NC* technique used in this paper creates an (L, M) data redundancy. It produces M coded data messages from the set of L original ones, so that L of them suffice for reconstructing the original messages, where $L < M$.

Let $v_i = (v_{1,i}, v_{2,i}, \ldots, v_{L,i})$, where $i \in \{1, 2, \ldots, M\}$, denote the encoding vectors associated with the coded packets y_i and M is the number of relay nodes. The set of the coded data messages is generated through the following system of equations:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{pmatrix} = \begin{pmatrix} v_{1,1} & v_{1,2} & \dots & v_{1,L} \\ v_{2,1} & v_{2,2} & \dots & v_{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M,1} & v_{2,M} & \dots & v_{M,L} \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{pmatrix} \quad (2)$$

The *BS* will need L data coded messages out of M to retrieve the original ones. It will proceed to solve the linear equation system that should be contained in the set of coded received packets $(y_{q_1}, y_{q_2}, \dots, y_{q_L})$, extracted from the initial constructed coded packets $(y_1, y_2, \dots, y_M)$, using the L corresponding encoding vectors, as follows:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{pmatrix} = \begin{pmatrix} v_{1,q_1} & \dots & v_{1,q_L} \\ v_{2,q_1} & \dots & v_{2,q_L} \\ \vdots & \ddots & \vdots \\ v_{L,q_1} & \dots & v_{L,q_L} \end{pmatrix} \times \begin{pmatrix} y_{q_1} \\ y_{q_2} \\ \vdots \\ y_{q_L} \end{pmatrix} \quad (3)$$

This *BS* can achieve this process with a simple elimination of Gauss or by inverting the matrix made up of the different rows $(q_1, \dots, q_L)$ of the matrix of encoding vectors used by the L out of M existing relay nodes.

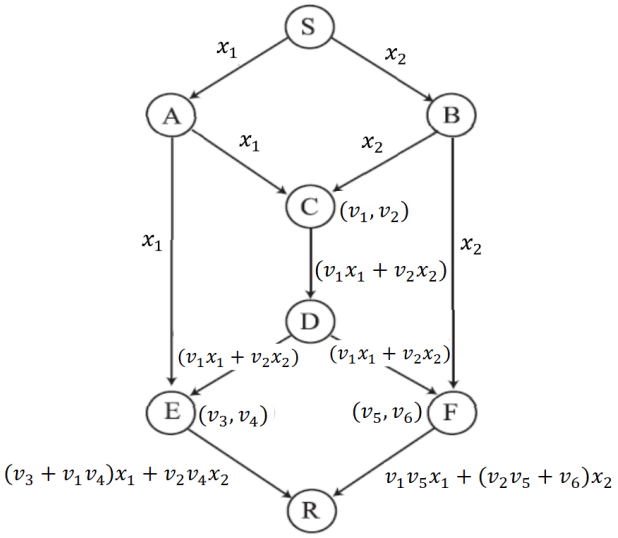

Figure 2. An example of linear network coding.

3.2. Assumptions about the Network and Sensor Nodes

Our scheme makes the following assumptions.

- The network has a hierarchical topology. Sensor nodes are grouped in clusters. Each cluster is headed by a cluster head (*CH*) selected at the beginning of the protocol by the base station (*BS*).
- The clustering is static and determined by a centralized K-means algorithm, which provides optimized choices of cluster heads and their member nodes that favor minimized distances between them.
- Member nodes of each cluster are responsible for detecting external events. They are called source nodes. *CH*s are the nodes in charge of forwarding received data to the *BS*. They are the ones susceptible to being compromised.
- Each sensor node is equipped with a unique matrix of randomly chosen integers of large size $(l \times L)$ considered as a secret key, where $l > t$ and $L > p$. These matrices are pre-loaded before the network deployment and are shared with the *BS*.
- The *BS* cannot be compromised and has essentially unlimited processing power compared to sensor nodes. Moreover, the *BS* is aware of the positions of all sensor nodes and their distances to each other in the network.

- The total number of *CH*s in the network is a minimum of p, chosen once by the user before deploying the network according to its size and scale, which are related to the network domain of application.
- Each source node has several paths to the *BS*. A node can send data to the *BS* through p different *CH*s. For simplicity, we assume that these *CH*s are of approximately similar distances to the source node.
- Each source node proceeds to send sensed data to the *BS* through the routing protocol after t sensed data measurements. In other words, each routing round ensures the transmission, for each source node, of a data message composed of t sensed original data packets.

3.3. Threat Model and Security Objectives

In the threat model, maliciousness is manifested in two ways. The first is when the malicious node refuses to cooperate in the routing process, i.e., it is a non-cooperative node that discards data packets once receiving them. The second is when the malicious node alters data packets before forwarding them or even injects a new packet with the same expected format. This type of node is known as a misbehaving node (*MN*). Since the next hop from source node in the path to the *BS* is always a *CH* node, the *MN*, in our proposition, is a *CH* node. The adversary is modeled as a coalition of *CH* nodes varying from 1 to K (Figure 3). They can proceed to:

1. reject packets received from source nodes, and stop them from reaching the *BS*. This attack, considered as a form of *DoS* attack, causes loss of packets and degrades the performance of the network;
2. modify packets received from source nodes or inject fake ones into the network. This is known as a pollution attack where the *BS* receives polluted packets that cannot be separated from the original ones in a conventional standard routing.

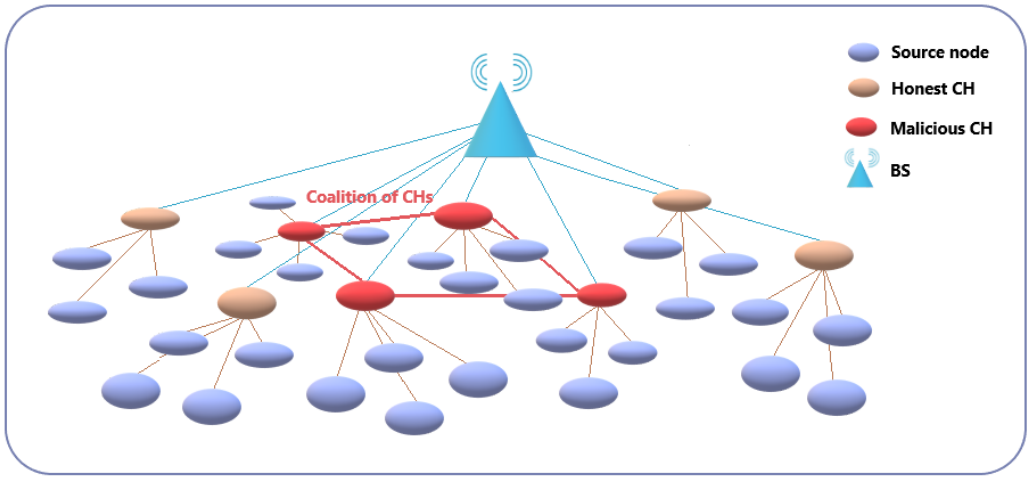

Figure 3. The network and adversary model.

The proposed solution guarantees that with a coalition of *MN*s of a size in the range of a threshold $K = p - t - 1$, the adversary fails to stop the successful reception of any message from a source node at the *BS*. The proposed solution is also protected against eavesdropping attacks as in [15]. These occur when an adversary has unauthorized access to useful data transmitted by any link in the network. The adversary, in this case, can be internal to the network as any relay node except the source nodes and the *BS*; or external to the network with the ability to wiretap inter-cluster or intra-cluster links.

Our proposition deals with the following security objectives:

- Availability: ensuring that messages transmitted by a source node reach the *BS*, thus providing survivability of the *WSN* services.

- Integrity: ensuring the reception of the correct original messages of source nodes when there are attempts by hackers to modify data en route to the *BS*.
- Detection and isolation: identifying adversarial behavior sources, and isolating adversary nodes from the routing to prevent any damage to the network.
- Confidentiality: ensuring secrecy of the wireless communication links to avoid eavesdropping.

4. Design and Implementation

This section provides the design of our new secure routing protocol, referred to as *SSP*. As described in Section 3.2, it applies to networks with hierarchical topology. Its main goal is to defend against *DoS* and pollution attacks and to detect the malicious *CH*s responsible for it through a modified *NC*-based routing. The protocol consists of three phases: (1) initialization; (2) *DoS* and pollution attack prevention; and (3) attack detection and punishment.

4.1. Initialization

The network is composed of randomly deployed sensor nodes and a *BS*. The solution we propose begins with clustering. The *BS* divides the network into N clusters, i.e., N groups of source nodes, headed by a *CH*. Obviously here, N must be greater than p since p is a sub-group of *CH* nodes. p is predefined by the user before deploying the network, taking into consideration the network size.

Next, the *BS* chooses, for each source node, a set of the nearest p *CH*s and sends their identities to it. As a result, the paths to be used for routing data elements from each source node are established and shared with the *BS*.
Data are transmitted from source nodes to *CH* nodes and then from *CH* nodes to the *BS*.

4.2. DoS and Pollution Attacks Prevention

Our solution main phase is the prevention against *DoS* attacks and pollution attacks, ensuring data availability and integrity despite the existence of misbehavior in the network. It prevents a coalition of malicious *CH*s executing a *DoS* attack by dropping all the packets they receive or achieving a pollution attack by modifying the packets before their transmission to the *BS*. It guarantees the reconstruction of original data messages at the *BS*. For this purpose, our protocol uses the *NC*-techniques as described in Section 3.1. It produces p parts from a particular data message of size t, where $t < p$, so that t of them suffice for reconstructing the original one at the final destination, the *BS*. This phase is composed of: (i) encoding and parts creation at the source node and (ii) decoding and parts reconstruction at *BS*.

4.2.1. Encoding and Parts Creation at Source Node

The encoding process is conducted by each source node N_i in the network. Each node is pre-equipped with its specific unique matrix. $A_i = [a_{k,q}]$ of large size $(l \times L)$ denotes the pre-loaded matrix, where a_k^q are A_i's integer coefficients, $k \in [\![1, l]\!]$ and $q \in [\![1, L]\!]$, chosen so that each combination of l columns forms an invertible matrix of size $(l \times l)$.

When measuring events, a node N_i registers the measurements in its memory. After t periods, it has collected t parts of these measurements in a vector of size t, $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,t})$, forming a data message of size t. Then, N_i randomly chooses a number p_i such that $l \leq p_i \leq l \times L$ and pinpoints the coefficient of A_i having the position p_i. Next, it extracts a sub-matrix $A_i' = [a_{k,q}]$ of size $t \times p$ starting at the indicated p_i position in the matrix A_i, where $k \in [\![1, t]\!]$ and $q \in [\![1, p]\!]$. Figure 4 shows the process of extraction of the sub-matrix A_i' used to generate the coded packets.

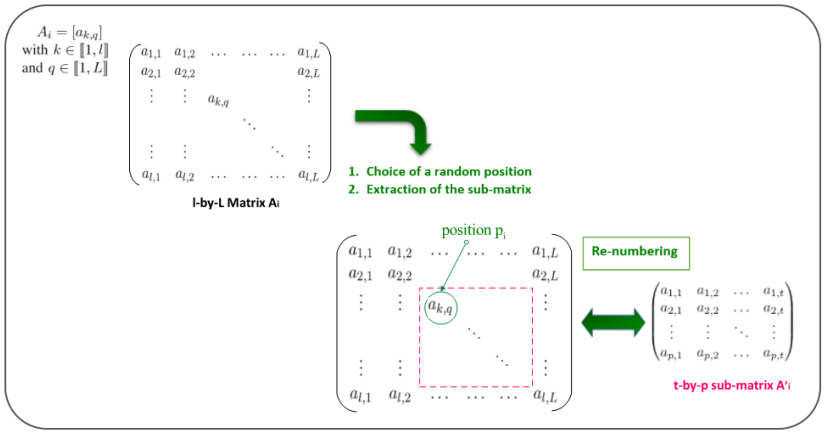

Figure 4. The sub-matrix A'_i extraction.

Next, N_i calculates p different parts from the t parts of the vector X_i. These parts are the elements of the vector $M_i = (m_{i,1}, m_{i,2}, \ldots, m_{i,p})$ of size p composed as follows: $M_i = A'_i \times X_i$: More specifically,

$$S_1 = \begin{cases} m_{i,1} = & a_{1,1}x_{i,1} + a_{1,2}x_{i,2} + \ldots + a_{1,t}x_{i,t} \\ m_{i,2} = & a_{2,1}x_{i,1} + a_{2,2}x_{i,2} + \ldots + a_{2,t}x_{i,t} \\ \vdots = & \vdots \\ m_{i,p} = & a_{p,1}x_{i,1} + a_{p,2}x_{i,2} + \ldots + a_{p,t}x_{i,t} \end{cases} \quad (4)$$

The value p_i, required to decode data at the *BS*, is attached to each data packet to be sent through a distinct *CH*.

After encoding the original data parts, N_i transmits these parts along the p distinct *CH*s, chosen at the beginning of the protocol. More specifically, N_i sends the packets (Id_i, $m_{i,j}$, p_i), where $j \in [\![1, p]\!]$, through the nearest p different *CH*s saved in its routing table, where Id_i is the node N_i identity and p_i is the random number chosen for conducting the network coding phase of N_i. Next, each *CH* forwards the received packets from multiple source nodes to the *BS*.

Each node N_i has p distinct paths to the *BS* through p different *CH*s. The scheme tolerates $(p - t - 1)$ of them being compromised. In other words, the coalition of misbehaving *CH*s involved in the attack here is assumed to be of a size of maximum K with $K = p - t - 1$ to ensure complete reception of all source nodes messages. In fact, the source node will have at least $t + 1$ parts received in the *BS*, to be sure it can rebuild the original measures. Beyond the threshold K, the performance of the protocol in term of availability and integrity decreases.

4.2.2. Decoding and Parts Reconstruction at *BS*

The decoding process is conducted by the *BS* for each source node. When the *BS* receives all the packets from *CH* nodes, it collects the packets belonging to the same source node N_i using its identity Id_i.

Since the adversary model guarantees that the *BS* receives at least $t + 1$ parts for each node, the *BS* is able to reconstruct the original data packet corresponding to each source node. In fact, the *BS* receives the parts $(m_{i,q_1}, m_{i,q_2}, \ldots, m_{i,q_{rp}})$ from different rp paths corresponding to different rp *CH*s, where $t \leq rp \leq p$ (rp represents the number of received packets for a source node). In this case, it triggers the decoding process as follows:

The *BS* checks for the p_i values received from the different packets, memorize the value p_i existing at least $t + 1$ times and distinguishes it as the true one (p_i^T) that has not been modified by an adversary. This is true because we assumed that the protocol guarantees that at least $t + 1$ packets are correctly received for each source node N_i. Next, it checks the pre-loaded matrix A_i and extracts from it the sub-matrix A'_i of size $t \times p$ at

the position value p_i^T. Afterwards, to reconstruct the original measurements, the *BS* uses each combination c of t data parts out of the rp received parts to resolve the system of equations $S_1(4)$. More specifically, it resolves a number value of C_{rp}^t systems to extract a set of possible original messages $X_i^{(c)} = (x_{i,1}^{(c)}, x_{i,2}^{(c)}, \ldots, x_{i,t}^{(c)})$ as follows, where $c \in \{1, \ldots, C_{rp}^t\}$ and $C_{rp}^t = \frac{rp!}{t! \times (rp-t)!}$:

$$\begin{cases} a_{q_{j_1},1} x_{i,1}^{(c)} + a_{q_{j_1},2} x_{i,2}^{(c)} + \ldots + a_{q_{j_1},t} x_{i,t}^{(c)} & = m_{i,q_{j_1}}^{(c)} \\ a_{q_{j_2},1} x_{i,1}^{(c)} + a_{q_{j_2},2} x_{i,2}^{(c)} + \ldots + a_{q_{j_2},t} x_{i,t}^{(c)} & = m_{i,q_{j_2}}^{(c)} \\ \vdots & = \vdots \\ a_{q_{j_t},1} x_{i,1}^{(c)} + a_{q_{j_t},2} x_{i,2}^{(c)} + \ldots + a_{q_{j_t},t} x_{i,t}^{(c)} & = m_{i,q_{j_t}}^{(c)} \end{cases}$$

where $\{q_{j_1}, q_{j_2}, \ldots, q_{j_t}\}$ are C_{rp}^t extracted combinations of size t from the set $\{q_1, q_2, \ldots, q_t, \ldots, q_{rp}\}$.

Next, the *BS* detects the most apparent message in the set of the constructed messages $X_i^{(c)}$ and recognizes it as the original one. The correct message will appear a number of times equal to C_{rp-pp}^t whereas each of the wrong messages will only appear once. pp depicts the number of polluted packets received at the *BS*.

The process of resolving the system of equations above can be done with a simple elimination of Gauss or by inverting the matrix made up of the different t encoding vectors of A_i' to obtain a matrix of t columns $q_{j_1}, \ldots, q_{j_t}$ corresponding to the different t *CH* nodes that sent the packets $m_{i,q_{j_1}}, \ldots, m_{i,q_{j_t}}$ and which are stored in the routing table at the *BS*. If the matrix A_i is chosen randomly and so is the matrix A_i', there is no known method to rebuild the original data, even partially, from $(t-1)$ parts.

4.3. Attack Detection and Punishment

Our solution then has an attack detection and punishment phase, which locates the misbehavior that affects the network availability or integrity. It identifies the *MN*s among the *CH*s executing the *DoS* or pollution attacks and applies a punishment mechanism to exclude them from the routing process. The *BS* is aware of all the nodes' locations and the different groups of clusters in the network. In addition, the *BS* has a control counter (*cc*) for each *CH*, which decides on its maliciousness.

At the end of each routing round, the *BS* is supposed to receive the set of packets $L_i = \{(Id_i, m_{i,j}, p_i)\}$ where $j \in \{1, 2, \ldots, p\}$ through p different paths coming from the nodes CH_{q_j}. q_j are the p *CH* identities saved in the routing table for each source node N_i. The *BS*'s decision depends on the source nodes and the packets received for each one. It begins by scanning the malicious *CH*s invoking a *DoS* attack then continues by scanning the malicious *CH*s effecting a pollution attack. Once it has collected the list of packets received for a source node N_i, the *BS* compares it to the list L_i.

First, for *DoS* attack detection, it notes the set of packets missing and identifies the *CH* identities $\{q_j\}_{j \in \{1,\ldots,p\}}$ in charge of transmitting them. Then, it increments their counter *cc* by 1, as they are suspected of being malicious nodes.

Second, for pollution attack detection and after collecting the list of packets received for a source node N_i, the *BS* proceeds via two steps:

1. Since it has already saved the true value p_i^T in its memory when calculating the original routed message, the *BS* checks for received packets containing a p_i value different from the p_i^T value. It recognizes the *CH*s that have sent these packets as potentially malicious nodes and increments their counter *cc* by 1.

2. It checks the C^{t}_{rp-pp} combinations of packets that generated the original message and notes the list of *CH* identities that have transmitted these packets. Then, it classifies the *CH*s with identities not belonging to this list as potentially being malicious and increments their *cc* by 1. The reason is that one polluted packet in a combination of t packets will generate a fake calculated message.

Lastly, if the *cc* of a specific *CH* node Id_{q_S} reaches the value r, where r is a threshold chosen by the user before deploying the network, the *BS* recognizes this *CH* as a selfish node. In other words, if the node with identity Id_{q_S} has effected a number of attacks, whether by refusing to send a data packet or by faking it, equal to r, the *BS* proceeds as follows:

- It excludes the node with identity Id_{q_S} from the network and bans it from the routing process by deleting it from the network nodes list.
- It invokes a network re-clustering using the updated list of existing nodes.
- It resets the paths used for each source node by choosing new p *CH* identities for each one.

In this way, the protocol increases the security level of the network, not only by preventing *DoS* and pollution attacks, but also by conducting active steps to localize their source and punish the *MN*s responsible.

4.4. Discussion

We present, in this subsection, some main points for discussing our solution in regards to the method used for ensuring the security aspects of availability and integrity, and confidentiality.

The main algorithm, the *NC* technique that creates a (t, p) data redundancy, is effective with respect to size; that is to say that rebuilding t measures only needs t parts whose total size is identical. It is, however, possible to use more parts $(p > t)$ to help protect the network against *DoS* attacks or pollution attacks. In other words, an attacker needs to compromise at least $p - t$ separate paths, i.e., in the worst case, $(p - t)$ *CH* nodes. If the attacker does not know the topology set up to route data in the network, it has no choice other than to randomly compromise the nodes. As a result, it will likely need to capture more than $(p - t)$ *CH* nodes.

This strong availability and integrity is obtained at the cost of extra data parts transmission for each source node, i.e., transmitting p parts instead of t parts where $p > t$. However, our approach does not require heavy operations: dispersal and reconstruction are computationally efficient [33] and the systems of equations have to be solved only by the *BS*.

It is not possible for a few compromised nodes to reconstruct the measures. In other words, there is no technique for rebuilding, even partially, the measurements from $t - 1$ parts. This is the major advantage of dispersing the information in the network.

The solution, based on *NC*, provides complete confidentiality, so that no information can be obtained by intercepting coded symbols or random values of p_i. In fact, a wiretapper cannot recover the original data packets of a source node N_i, whether it is an outsider node or a captured *CH* inside the network. The main reason is that the corresponding matrix A_i belonging to a source node N_i is secret and not communicated to the destination when routing data packets. Moreover, the chosen values of p_i for each node N_i are changing and chosen randomly each round, so no other node but the *BS* is able to reconstruct the original data. A detailed demonstration is available in our previous paper [15].

Apart from security requirements, our solution minimizes the overheads compared to conventional network coding systems. First, instead of transferring all the coded vectors, i.e., the coefficients $a_{k,q}$, where $k, q \in \{[\![1, t]\!], [\![1, p]\!]\}$, along with the coded p symbols, a source node N_i adds only a single digit p_i to the coded vector to be sent. Second, the solution performs encoding only at the level of source nodes; there is no re-coding at the level of intermediate nodes, i.e., the *CH* nodes. This decreases the energy consumption and increases the network lifespan.

5. Simulation Setup and Results

In this section, we provide a simulation setup where we introduce the general parameters used for the experiments, and simulation results where we present the different scenarios implemented to show the benefits of our proposed solution.

5.1. Simulation Setup

For our experiments, we assume a fixed number of static nodes that are randomly distributed in a large two-dimensional area and a high processing power *BS* located randomly within the nodes' geographic area. The nodes are generated through a uniform random distribution where all values of the interval have equal probability. We assume that all nodes have the same amount of energy, at the beginning of the protocol. The benefit of the protocol is tested through a series of simulations with *MATLAB* by varying the number of rounds and the number of misbehaving nodes (*MN*). For comparison purposes, *SSP* is compared with the *SNCR* protocol; the same confidential routing protocol but without incorporating the security scheme against *DoS* and pollution attacks. The benefit of the protocol is also tested by varying the size of the network, i.e., the number of total sensor nodes in the network, to verify its scalability.

The clustering phase is operated with a *K*-means algorithm for simplicity and efficiency. Both protocols use direct transmission from *CH*s to *BS* as a communication pattern. The simulations are performed into two phases. The first allows routing of packets from member nodes to *CH* nodes while the second phase processes the required computations within the *BS* and verifies which packets have been successfully recovered.

Data availability against selfish nodes is evaluated in terms of the parameters of the Packet Delivery Ratio (*PDR*) and correct messages ratio, whereas data integrity against pollution attackers is evaluated in terms of the number of correct messages received. Table 1 shows the input parameters for the experiments. We provide an example of the *BS* coordinates used in this simulation and illustrate it later, in the network presentation figures.

Table 1. Network parameters.

Parameter	Value
Size of the region	600 × 600 (in meters)
Placement of nodes	uniform random distribution
Location of the *BS*	x = 150, y = 200
Routing algorithm used	SSP and SNCR
Number of rounds	500
Number of nodes deployed	100
Number of clusters *N*	15
Total number of original messages *O*	85
number of original packets *t*	5
number of transformed packets *p*	10
number of *MN*s among the *CH*s (threshold *K*)	4
Initial energy of a sensor node (E_0)	2 J
E_{elec}	50 nJ/bit

5.2. Simulation Results

The sensor network, which is composed of 100 nodes with IDs [1; 100] randomly deployed, is illustrated in Figure 5.

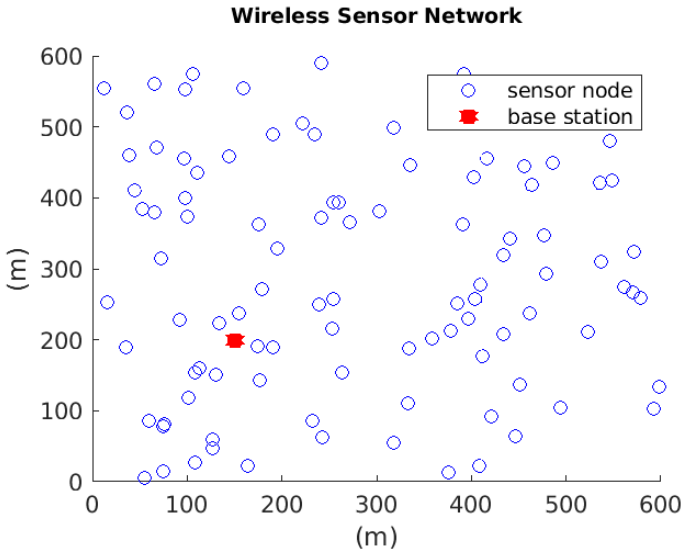

Figure 5. The initially deployed network.

We execute a *K*-means algorithm, and the obtained result is 15 clusters distributed as shown in Figure 6. The *K*-means algorithm processes an unsupervised learning with the aim of minimizing the distance between member nodes and their *CH*s. Consequently, it reduces the energy consumption of inter-cluster communication.

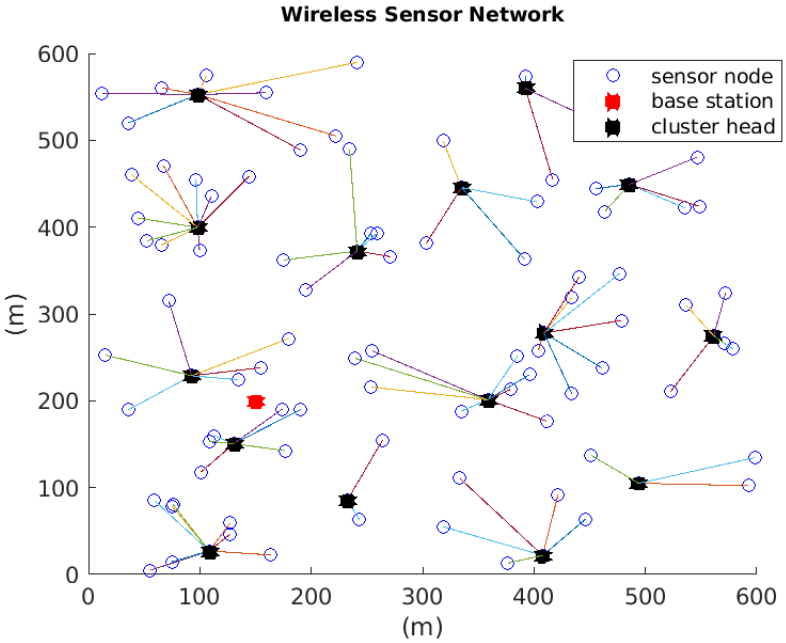

Figure 6. The network clustered.

Figure 7 depicts the *PDR* ratio and the correct messages ratio for both *SNCR* and *SSP* when a *DoS* attack is performed. The first ratio is between the number of data packets received at the *BS* and the number of data packets sent in the network. The second ratio is between the number of successfully constructed messages within the *BS* and the number of total messages sent by source nodes.

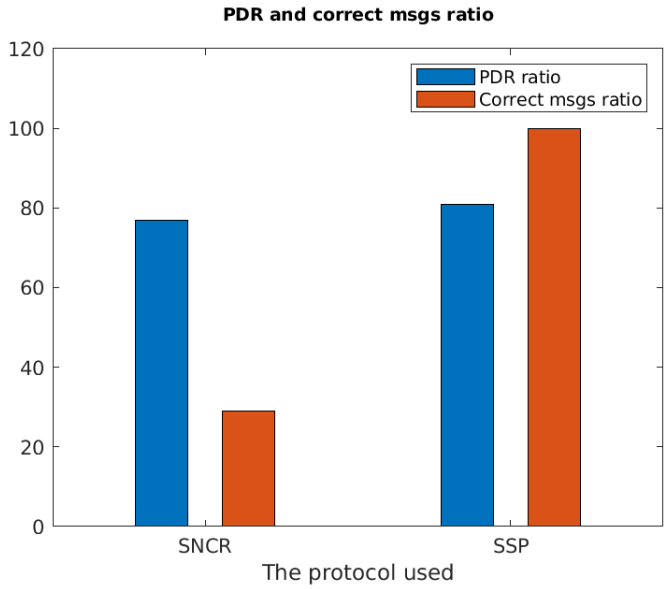

Figure 7. *PDR* and correct messages ratio.

Calculations were done by varying the number of rounds and fixing the number of selfish *CH* nodes at the value $t - p - 1$. The resulting values were similar for each round for each protocol. This is why we presented a single value for the *PDR* ratio and for the correct messages ratio for both protocols.

Figure 7 shows that in the case of a defenseless *SNCR*, the value of the correct messages ratio successfully received at the *BS* decreased significantly compared to the *PDR* ratio value. It also shows that for the *SSP* protocol, the value of the correct messages ratio is greater than the *PDR* ratio value. This graph shows that a *DoS* attack conducted by the *MN*s and affecting the *PDR* ratio parameter has a negative influence on the total number of successfully received messages for the *SNCR* protocol, which decreased to 30%. It does not affect the total number of successfully received messages in the case of the *SSP* protocol, which maintains a value of almost 100%. In fact, as the *PDR* ratio decreases, the correct messages ratio increases. We can conclude from these results that the proposed solution is beneficial in maintaining reception of a high number of correct data packets at the final destination despite an adversarial attack. In fact, if a number of data parts are lost due to attacks on the network, the *BS* is still capable of constructing all the original data packets.

Figure 8 presents the impact of increasing the number of malicious *CH* nodes, performing a pollution attack, on the variation in the number of correct messages received at the *BS* for *SNCR* and *SSP* protocols. The number of correct messages received at the *BS* is the number of correctly constructed messages at the end of the routing process. This variation decreases significantly as the intensity of pollution attack increases in the case of *SNCR*, as shown by the red line in Figure 8. As can be seen in the graph, the presence of two or three misbehaving *CH* nodes in the network reduced the number of correct messages received at the *BS* by more than half. This number continued to decrease until it reached 0 when the number of *MN*s was 7. This same number of correct messages has a certain stability in the case of the *SSP* protocol, when the intensity of the pollution attack increases. From the same graph, it is clear that the number of correct messages is higher in *SSP* compared to *SNCR*, as shown by the blue line. From point 0 to point 6 on the x-axis where the number of *MN*s goes from 0 to 6, the number of correct messages is constant and equal to $O = 85$ (total number of source nodes of original messages to be sent from source nodes), i.e., all the messages sent from source nodes are successfully received at the *BS*. Then, as more *MN*s are present, the number of correct messages starts decreasing and reaches a value of 0 only when the number of misbehaving *CH* nodes is equal to 10.

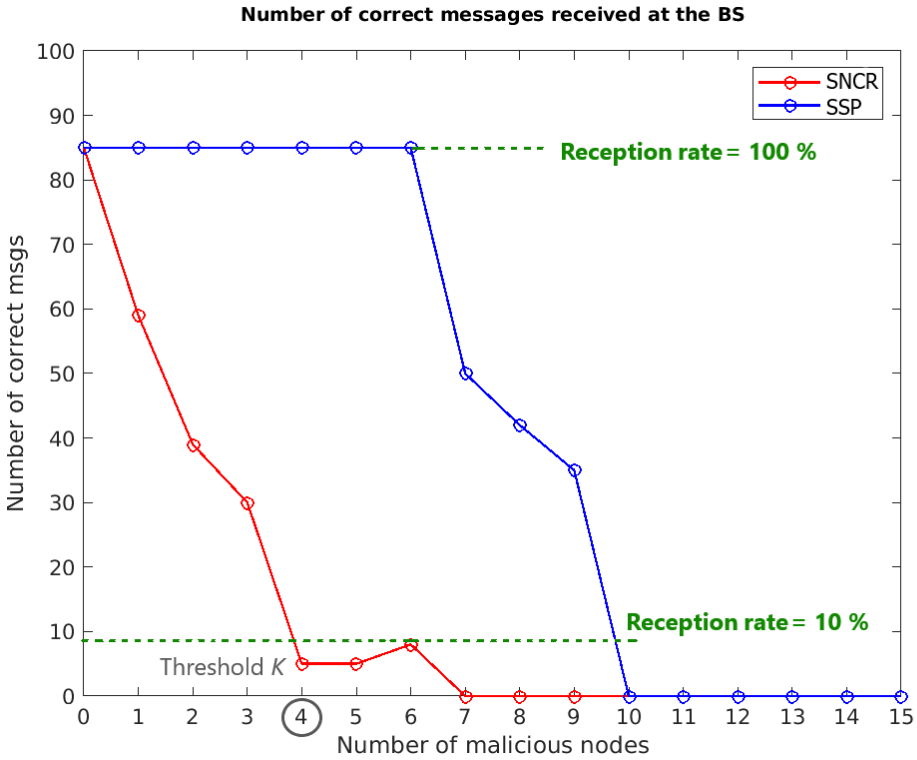

Figure 8. Number of correct messages for *SSP*.

This is because for the interval [0, 4], where the number of malicious *CH* nodes is less than the threshold $K = 4(K = p - t - 1)$ as assumed previously, the remaining number of honest (*CH*) nodes is at least $t = 5$. This guarantees that the number of correct parts transmitted for each source node is at least $t + 1$ parts, which is a sufficient number to reconstruct the original data parts. Then, for the interval beyond 4, where the number of malicious *CH* nodes is greater than the threshold $K = 4$, the number of collected parts starts going below t for a member node. The number of correct messages is equal to O in some cases in this interval, when the number of malicious *CH*s is 5 and 6. This is explained by the fact that the total number of *CH* nodes (15 here) in the network is greater than the limit $p = 10$. Therefore, when the network is more scalable, the tolerance for choosing the threshold value K is enhanced.

Figure 9 demonstrates the scalability aspect of the protocol. It depicts the variation in the percentage of correct messages received at the *BS* when a *DoS* attack is executed, related to the size of the network, i.e., the total number of sensor nodes in the network and the number of clusters in it. We chose for this simulation a number of malicious *CH* nodes that belonged to the interval beyond the threshold $K = p - t - 1$. K is set here to 6. It can be seen that the percentage of correct messages increases as the network size increases. In fact, when the number of sensor nodes increases, the number of clusters increases. Consequently, the number of *CH* nodes increases, as does the number of possible paths for a source node to route data packets to the *BS*. In this case, despite S, the number of *MN*s is greater than the threshold $K =$, the probability of collecting $t + 1$ parts is, obviously, raised. The reason is that S is distributed among a higher number of honest *CH*s even with a number of *MN*s S greater than the threshold K. The protocol enhances tolerance to the existence of *MN*s in the network while guaranteeing greater prevention against a *DoS* attack when the network gets larger. By evaluating the scalability factor, we can conclude that the proposed solution is capable of supporting network expansion without degrading the security performance.

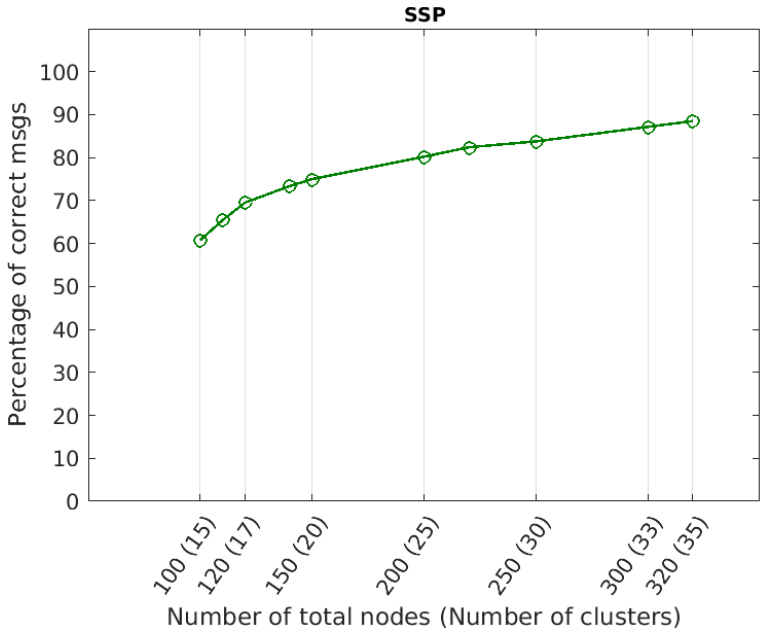

Figure 9. The network clustered.

6. Conclusions

In this study, we designed a scheme to prevent misbehaving nodes influencing network availability and integrity, using the *NC* technique and data redundancy creation at source nodes. The approach integrates a module to detect these malicious nodes and executes a punishment mechanism to exclude them from the routing process. It also ensures confidential data routing in the presence of eavesdroppers.

Our approach was implemented in a cluster-based network exposed to a *DoS* attack and pollution attack using *MATLAB*. These attacks were generated by a coalition of selected misbehaving *CH* nodes. They dropped the packets received from source nodes on their way to the *BS* when executing a *DoS* attack, and they incorrectly altered the packet in transit or injected a new fake one into the network when executing a pollution attack. The protocol that we have designed improves the percentage of correct messages received as compared to a defenseless *SNCR* protocol. The *BS* successfully rebuilds the original data from a source node despite selfish behavior that diminishes the number of packets received compared to the number of packets sent. In addition, it ensures a complete and correct reception of all original packets sent by source nodes with a number of malicious *CH* nodes initiating a pollution attack and varying in the range of a certain threshold *K*. The *BS* successfully reconstructs the original data of a source node even when the adversary alters the packets in transit before being sent to the *BS*. Moreover, the protocol favors scalability. It reduces the percentage of lost packets as the size of the network and so the number of malicious nodes increases.

Author Contributions: Conceptualization, H.R. and K.T.; methodology, H.R.; software, H.R.; validation, D.S., K.T., R.A. and S.G.; formal analysis, H.R. and D.S.; writing—original draft preparation, H.R.; writing—review and editing, D.S.; supervision, R.A. and S.G.; funding acquisition, D.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the Région Nouvelle-Aquitaine under the grant for project "SVP-IoT" and by the MIRES research federation under grants for projects "CANIoT" and "SECIoT".

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

Abbreviations

The following abbreviations are used in this paper:

N_i	A source node belonging to a cluster
CH	Cluster head
BS	Base station
A_i	Matrix of size $l \times L$ unique for each source node N_i and pre-loaded before network deployment
$a_{k,q}$	Value of the matrix A_i at the kth line and qth column
A'_i	Sub-matrix of size $t \times p$ from the matrix A_i
p_i	Integer value chosen randomly at the beginning of each sensor node to ensuring confidentiality
Id_i	Sensor node identity
t	Number of original measurements sensed by each source node
p	Number of packets produced from the t original ones for each source node N_i
K	Threshold of number of selfish CH nodes tolerated with $K = p - t - 1$
X_i	Original data message of size t to be sent to BS for each source node N_i
$x_{i,j}$	jth packet element of the message X_i
S_1	System of equations used to ensure data confidentiality and availability
M_i	Data message of size p generated from the system of equations S_1 for each source node N_i
$m_{i,j}$	jth packet element of the message M_i
rp	Number of packets received at the BS for a node N_i
pp	Number of polluted packets received at the BS for a node N_i
cc	Control counter specific to each CH

References

1. Fu, C.; Jiang, Z.; Wei, W.; Wei, A. An energy balanced algorithm of LEACH protocol in WSN. *Int. J. Comput. Sci. Issues (IJCSI)* **2013**, *10*, 354.
2. Liu, X. A survey on clustering routing protocols in wireless sensor networks. *Sensors* **2012**, *12*, 11113–11153. [CrossRef] [PubMed]
3. Boyinbode, O.; Le, H.; Takizawa, M. A survey on clustering algorithms for wireless sensor networks. *Int. J. Space-Based Situated Comput.* **2011**, *1*, 130–136. [CrossRef]
4. Mostafaei, H.; Montieri, A.; Persico, V.; Pescapé, A. A sleep scheduling approach based on learning automata for WSN partialcoverage. *J. Netw. Comput. Appl.* **2017**, *80*, 67–78. [CrossRef]
5. Seah, W.K.; Eu, Z.A.; Tan, H.P. Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP)-Survey and challenges. In Proceedings of the 2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, Aalborg, Denmark, 17–20 May 2009; pp. 1–5.
6. Huang, X.; Ahmed, M.; Sharma, D. Protecting from inside attacks in wireless sensor networks. In Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, Sydney, Australia, 12–14 December 2011; pp. 186–191.
7. Ahmed, M.R. Protecting Wireless Sensor Networks from Internal Attacks. Ph.D. Thesis, University of Canberra, Canberra, Australia, 2014.
8. Hussain, A.; Heidemann, J.; Papadopoulos, C. A framework for classifying denial of service attacks. In Proceedings of the 2003 conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Karlsruhe, Germany, 25–29 August 2003; pp. 99–110.
9. Jhaveri, R.H.; Patel, S.J.; Jinwala, D.C. DoS attacks in mobile ad hoc networks: A survey. In Proceedings of the 2012 Second International Conference on Advanced Computing & Communication Technologies, Rohtak, India, 7–8 January 2012; pp. 535–541.
10. Tayebi, A.; Berber, S.; Swain, A. Wireless Sensor Network attacks: An overview and critical analysis. In Proceedings of the 2013 Seventh International Conference on Sensing Technology (ICST), Wellington, New Zealand, 3–5 December 2013; pp. 97–102.
11. Osanaiye, O.A.; Alfa, A.S.; Hancke, G.P. Denial of service defence for resource availability in wireless sensor networks. *IEEE Access* **2018**, *6*, 6975–7004. [CrossRef]
12. Gavric, Z.; Simic, D. Overview of DOS attacks on wireless sensor networks and experimental results for simulation of interference attacks. *Ing. Investig.* **2018**, *38*, 130–138. [CrossRef]
13. Mamidwar, N.V.; Gothawal, D. Schemes against Pollution Attack in Network Coding: A Survey. *Int. J. Comput. Sci. Inf. Technol.* **2015**, *6*, 5085–5089.
14. Gemeda, K.A.; Gianini, G.; Libsie, M. The effect of node selfishness on the performance of WSN cluster-based routing algorithms. In Proceedings of the AFRICON 2015, Addis Ababa, Ethiopia, 14–17 September 2015; pp. 1–5.

15. Rhim, H.; Abassi, R.; Tamine, K.; Sauveron, D.; Guemara, S. A Secure Network Coding-Enabled Approach for a Confidential Cluster-Based Routing in Wireless Sensor Networks. In Proceedings of the SAC '20: 35th Annual ACM Symposium on Applied Computing, Brno, Czech Republic, 30 March–3 April 2020; pp. 2151–2157. [CrossRef]
16. Kumar, V.D.; Navaneethan, C. Protection against denial of service (dos) attacks in wireless sensor networks. *Int. J. Adv. Res. Comput. Sci. Technol.* **2014**, *2*, 439–443.
17. Mansouri, D.; Mokddad, L.; Ben-Othman, J.; Ioualalen, M. Preventing denial of service attacks in wireless sensor networks. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 3014–3019.
18. Fouchal, S.; Mansouri, D.; Mokdad, L.; Iouallalen, M. Recursive-clustering-based approach for denial of service (DoS) attacks in wireless sensors networks. *Int. J. Commun. Syst.* **2015**, *28*, 309–324. [CrossRef]
19. Das, S.K.; Saha, B.J.; Chatterjee, P.S. Selfish node detection and its behavior in WSN. In Proceedings of the Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Hefei, China, 11–13 July 2014; pp. 1–6.
20. Virmani, D.; Hemrajani, M.; Chandel, S. Exponential trust based mechanism to detect black hole attack in wireless sensor network. *arXiv* **2014**, arXiv:1401.2541.
21. Kalkha, H.; Satori, H.; Satori, K. Preventing black hole attack in wireless sensor network using HMM. *Procedia Comput. Sci.* **2019**, *148*, 552–561. [CrossRef]
22. Yao, S.; Chen, J.; Du, R.; Deng, L.; Wang, C. A survey of security network coding toward various attacks. In Proceedings of the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, Beijing, China, 24–26 September 2014; pp. 252–259.
23. Bassoli, R.; Marques, H.; Rodriguez, J.; Shum, K.W.; Tafazolli, R. Network coding theory: A survey. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 1950–1978. [CrossRef]
24. Talooki, V.N.; Bassoli, R.; Lucani, D.E.; Rodriguez, J.; Fitzek, F.H.; Marques, H.; Tafazolli, R. Security concerns and countermeasures in network coding based communication systems: A survey. *Comput. Netw.* **2015**, *83*, 422–445. [CrossRef]
25. Adeli, M.; Liu, H. Secure network coding with minimum overhead based on hash functions. *IEEE Commun. Lett.* **2009**, *13*, 956–958. [CrossRef]
26. Kim, Y.S. Refined secure network coding scheme with no restriction on coding vectors. *IEEE Commun. Lett.* **2012**, *16*, 1907–1910. [CrossRef]
27. Yu, Z.; Wei, Y.; Ramkumar, B.; Guan, Y. An efficient signature-based scheme for securing network coding against pollution attacks. In Proceedings of the IEEE INFOCOM 2008-The 27th Conference on Computer Communications, Phoenix, AZ, USA, 13–18 April 2008; pp. 1409–1417.
28. SadrHaghighi, S.; Khorsandi, S. An identity-based digital signature scheme to detect pollution attacks in intra-session network coding. In Proceedings of the 2016 13th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), Tehran, Iran, 7–8 September 2016; pp. 7–12.
29. Liu, X.; Huang, J.; Wu, Y.; Zong, G. A privacy-preserving signature scheme for network coding. *IEEE Access* **2019**, *7*, 109739–109750. [CrossRef]
30. Li, Y.; Zhang, F.; Liu, X. Secure Data Delivery with Identity-based Linearly Homomorphic Network Coding Signature Scheme in IoT. *IEEE Trans. Serv. Comput.* **2020**. [CrossRef]
31. Li, S.Y.; Yeung, R.W.; Cai, N. Linear network coding. *IEEE Trans. Inf. Theory* **2003**, *49*, 371–381. [CrossRef]
32. Lima, L.; Médard, M.; Barros, J. Random linear network coding: A free cipher? In Proceedings of the 2007 IEEE International Symposium on Information Theory, Nice, France, 24–29 June 2007; pp. 546–550.
33. Rabin, M.O. The information dispersal algorithm and its applications. In *Sequences*; Springer: Berlin/Heidelberg, Germany, 1990; pp. 406–419.