sensors

MDPI

Article

A Data Clustering Algorithm for Detecting Selective Forwarding Attack in Cluster-Based Wireless Sensor Networks

Hao Fu †, Yinghong Liu †, Zhe Dong and Yuanming Wu *

School of Optoelectronic Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China; 201722050606@std.uestc.edu.cn (H.F.); liuyinghong@std.uestc.edu.cn (Y.L.); 201722050603@std.uestc.edu.cn (Z.D.)

* Correspondence: ymwu@uestc.edu.cn; Tel.: +86-138-0807-5543

† Both authors contribute equally as co-first authors.

Received: 5 November 2019; Accepted: 17 December 2019; Published: 19 December 2019

Abstract: In cluster-based wireless sensor networks, cluster heads (CHs) gather and fuse data packets from sensor nodes; then, they forward fused packets to the sink node (SN). This helps wireless sensor networks balance energy effectively and efficiently to prolong their lifetime. However, cluster-based WSNs are vulnerable to selective forwarding attacks. Compromised CHs would become malicious and launch selective forwarding attacks in which they drop part of or all the packets from other nodes. In this paper, a data clustering algorithm (DCA) for detecting a selective forwarding attack (DCA-SF) is proposed. It can capture and isolate malicious CHs that have launched selective forwarding attacks by clustering their cumulative forwarding rates (CFRs). The DCA-SF algorithm has been strengthened by changing the DCA parameters (Eps, Minpts) adaptively. The simulation results show that the DCA-SF has a low missed detection rate of 1.04% and a false detection rate of 0.42% respectively with low energy consumption.

Keywords: data clustering algorithm; selective forwarding attack; cumulative forwarding rate; cluster-based WSN

1. Introduction

A wireless sensor network (WSN) is a self-organizing network formed by a mass of small and cheap sensor nodes, which have low energy, poor computing ability, and small storage. The cluster-based WSN has been widely applied in large-scale data gathering WSNs [1,2]. In the dense cluster-based WSN, as shown in Figure 1, member nodes (MNs) send data packets to their cluster heads (CHs). Then, CHs forward these packets to the next-hop CHs until they reach the sink node (SN). In this way, each CH does not have to exchange data with the SN directly. On the one hand, the direct communication between the CH and SN may fail due to the long-distance or poor channels. On the other hand, in some conditions, the energy cost of multi-hop communication is less than that of direct long-distance communication. All the nodes in the network take turns to act as CHs, so the energy consumption can get balanced. The network lifetime also becomes longer.

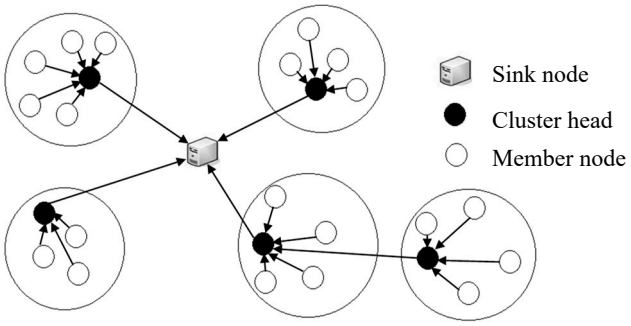

Figure 1. An example of a cluster-based wireless sensor network (WSN).

CHs may usually suffer various attacks launched inside or outside, due to their vital roles in the cluster-based WSN. The selective forwarding attack is one of the most common attacks, where a CH compromised by the attacker drops all or part of the data packets. The severe influence includes not only high data loss and poor quality of service (QoS) but also the damage to energy-balanced routing protocols [3,4].

Selective forwarding attacks are divided into black hole attack, gray hole attack, and on/off attack. Black hole attack is the worst selective forwarding attack, since a malicious node drops all the packets it should forward [5]. In a gray hole attack, a malicious node randomly drops packets [6]. On/off attack is where a malicious node periodically drops all or part of data packets only within a time interval and acts normally in the other time [7]. In the selective forwarding attack, both the number and rounds of packets dropping are random, which is nearly impossible to be distinguished from that caused by poor channel quality [8]. This makes it easy for malicious nodes to hide their identities and increases the difficulty of detecting selective forwarding attacks. If a malicious node cannot be detected and isolated quickly, it would continue dropping data packets. Meanwhile, any node sentenced as a malicious node will be deprived of the rights owned by normal nodes, including participating in packet forwarding. The misdetection of normal nodes makes the number of data forwarding nodes in the network decrease, and the energy consumption of a single node increases. Consequently, the network lifetime will be shortened. In summary, under low energy consumption, the goals of detecting selective forwarding attacks [9] are as follows: low missed detection rate (MDR), low false detection rate (FDR), and high detection speed.

Most detection schemes on selective forwarding attacks employ neighbor monitoring nodes and their reputations without taking the impact of channel quality on detection into account [7–13]. Some detection schemes take advantage of the clustering algorithm [14], but the channel quality is not included in the data set collection process. This causes a normal node with a lower cumulative forwarding rate (CFR) to be misjudged as a malicious node due to a poor channel.

A poor channel affects the current forwarding rates and CFRs heavily. If the channel quality is ideal, the forwarding rate always maintains 100%; otherwise, the data packets have to be dropped. If the scheme does not consider the impact of channel quality on the CFR, it cannot distinguish who has made data packets dropped, malicious nodes or poor channels. So, it is difficult for a WSN to detect selective forwarding attacks. Most schemes assume the perfect channel and have a good performance, but in practice, they result in a low correct detection rate where a poor channel exists.

The motivation for utilizing the data clustering algorithm (DCA) is to find out anomalous CFRs of malicious nodes under the same channel with normal nodes. If a node is compromised and becomes malicious, its CFR will be anomalous to that of a normal node. Our scheme introduces one DCA based on density to distinguish the anomalous CFRs. It divides CFRs into different groups of similar ones. In the density-based clustering DCA, it doesn't need to set the number of clusters in advance, and it can divide the CFRs into proper clusters depending on the distribution of the CFRs. The time and space complexity of a DCA can reduce to a low level so that it can be performed in sensor nodes.

In this paper, Section 2 reviews related works to help understand the proposal of our scheme. Section 3 elaborates on the proposed DCA for detecting a selective forwarding attack (DCA-SF) scheme. Section 4 conducts some simulation experiments, analyzes the results, and evaluates the advantages of DCA-SF. Section 5 gives conclusions and future work.

2. Related Work

2.1. Schemes against Selective Forwarding Attack

Since the selective forwarding attack was first proposed by Karlof [10], many schemes for detecting it have come out.

Semantic and Abhijit [11] proposed a selective forwarding attack detection scheme under the classical clustering algorithm, LEACH (Low Energy Adaptive Clustering Hierarchy). A counter is equipped in the base station. The counter checks whether the CH receives the data packets correctly and whether data packets reach the base station. A node is judged by its reliable neighbor nodes depending on selective time-variant flooding tests. The node will be charged as malicious once it fails the tests. In this scheme, the base station detects CHs. However, this scheme can only work in a small-scale cluster-based WSN. As the nodes increase, more data packets than ever would be sent to some nodes in the flooding test. Finally, too much test energy consumption itself can bring severe results to the whole network. In addition, ignoring the influence of the channel quality, this scheme often regards normal nodes as malicious under poor channel quality.

The watchdog [15] can monitor neighbor nodes so well that it is often available to detect selective forwarding attacks in WSNs. The node uses the watchdog to monitor neighbor nodes whether they forward its data packets or not. As shown in Figure 2, the watchdog supervisor deployed on node A detects whether neighbor nodes forward packets from A.

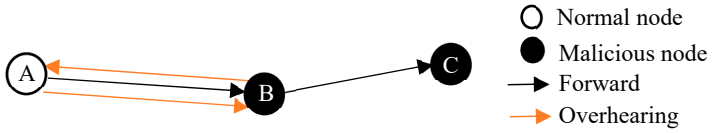

Figure 2. Watchdog monitoring mechanism.

The watchdog mechanism may fail to work correctly when both B and C are malicious nodes. This time C may launch an attack, as shown in Figure 2. B forwards the data packet sent by A to C. In this case, A's watchdog monitor will detect that B has forwarded the packets from A, and will not regard B as a malicious node. However, if C drops A's packets sent by B, the watchdog mechanism can not detect this event [16]. Schemes proposed in the literatures [8,17] have improved the watchdog mechanism. These schemes assign inspector nodes and cooperative nodes to monitor the relay nodes. The scheme in [8] provides a novel network model that will also be adopted in our scheme. In a cluster, there are three types of nodes: MN, CH, and inspector node (IN). The IN supervises whether the CH forwards the data packets from MNs in the cluster. Each time the CH and IN are replaced, IN communicates with the neighbor IN to prevent the collusion attack shown in Figure 2. The reputation is also introduced when judging whether the CH is a malicious node or not. However, in the scheme of [8], if both the CH and IN become malicious, their MNs cannot inform other clusters of the message that "the CH and IN are malicious". In this case, CH and IN continue to collude and launch attacks, causing packets of MNs in the cluster to fail to forward. The scheme [17] adds a cooperative node to monitor nodes that forward packets based on the watchdog mechanism. Unfortunately, the cooperative node may become malicious. The scheme in [12] improves the detection against such collusion attacks. When the MNs in one cluster find that the IN ignores the attack behavior of the CH, they will isolate the CH and the IN, and elect a new CH and IN. However, the channel quality problem is not considered in the scheme. Poor channels will lower the reputation of a normal CH, which increases the probability of misjudging a normal node as a malicious node.

In recent years, machine learning algorithms have been increasingly used in various research fields. If machine learning algorithms are applied in detecting attacks in WSNs, it will be good for detecting cunning attack patterns. However, the scheme has to take its time complexity and space complexity into account in WSNs because of the small memory space and the limited energy of sensor nodes. The scheme in [14] uses the DCA E-DBSCAN [18] and random forest algorithm [19] to detect network attacks of WSN. In the scheme, the data set organized by data from CHs to a SN is clustered by E-DBSCAN in SN. By clustering data, SN can pick up the anomalous data points to distinguish malicious nodes from normal ones. E-DBSCAN is easy to be used in WSN for detecting network attacks. It can distinguish the abnormal data from the normal data. Unfortunately, it cannot differentiate malicious CHs from normal CHs. In the random forest algorithm, the training set used for off-line training is the KDD Cup 99 data set [20] composed of nearly five million 41-dimensional vectors, which is a classical training set used for computer network attacks detection. However, gathering these vectors will be a big overhead for a WSN.

2.2. Data Clustering Algorithm (DCA)

The DCA, which is known as unsupervised classification, can divide a data set into many clusters according to their similarities. Several different DCAs have been proposed. CURE [21] and BIRCH [22] are based on hierarchy. CURE uses representative points as central points in the cluster. This can identify a variety of complex shapes and clusters of different sizes. However, in our scheme, the number of samples in the data set is relatively small, so it is difficult to find representative points. Furthermore, the complexity of CURE is $O(m^2)$ for low dimensions. This is too high to be performed in the sensor node. The basic idea of BIRCH is that the importance of each data point must be different so that different data samples can be treated differently. The BIRCH algorithm has a unique data structure called a clustering feature tree (CF-TREE). However, if the distribution cluster of the data set is not convex, the clustering effect is not good enough. Spectral clustering [23] based on graph theory is widely used in image analysis. However, how to automatically determine the number of clusters has become one of the key problems to be solved in spectral clustering. STING [24] is a grid-based multi-resolution clustering technique that divides a spatial region into rectangular cells. This technique has a fast processing speed but may reduce the quality and accuracy of the cluster.

K-means [25] and K-medoids [26] are based on partitioning. In K-means and K-medoids, clusters are groups of data that are characterized by a small distance to the cluster center [27]. K-means divides a data set into K clusters, and it make points in the same cluster have high similarity and low similarity in different clusters. The center of one cluster is the mean of all the points in the cluster. Every point in the data set must be divided into one cluster based on the shortest distance to the cluster center. The time complexity of K-means is O(m), where m is the number of points in the data set. This is an acceptable complexity in sensor nodes. However, the value of K needs to set in advance, and the choice of initial centers is random. These are two disadvantages of K-means. To overcome these two disadvantages, some methods are proposed. In the literature [28], the proposed method sets a K value and initial centers by rival penalized competitive learning (RPCL) [29]. However, the time complexity is $O(m^2)$, which is too large to be performed in sensor nodes. Kaufman et al. [30] proposed a heuristic method of estimating the local density of data points to pick up initial cluster centers of K-means. Furthermore, Dhillon et al. [31] recalculated cluster centers during the iteration to optimize clustering performance. However, the time complexity or space complexity is more than $O(m^2)$, so they are not suitable to be performed in sensor nodes. K-medoids is similar to K-means. The most significant difference is that the center of one cluster in K-medoids is a point rather than the mean of points in the cluster as in K-means. This improvement avoids the impact of isolated points on the mean. Traditional K-medoids is based on the partitioning around medoids (PAM) [32]. The time complexity of K-medoids is $O(m^2)$. Furthermore, it also needs to set the value of K in advance. So, this algorithm is not suitable to be performed in sensor nodes.

Density peaks clustering (DPC) [27] based on density is a recently proposed clustering algorithm published in *Science* magazine. This algorithm is based on two assumptions: that cluster centers are characterized by a higher density than their neighbors and by a larger distance from points with higher densities. It is performed clustering in terms of each data point's ρ value and δ value. For the data point i, the ρ value and δ value of point i are shown in Equations (1) and (2).

$$\rho_i = \sum_{j \neq i} \chi\left(d_{ij} - d_c\right) \tag{1}$$

$$\delta_i = \min_{j:\rho_j > \rho_i} \left(d_{ij}\right) \tag{2}$$

The cutoff distance d_c is a preset parameter, and its value varies in different methods. d_{ij} is the Euclidean distance between point i and point j, and $\chi(x) = 1$ when $x < 0$; otherwise $\chi(x) = 0$. That means that $\chi(d_{ij} - d_c) = 1$ when the distance d_{ij} between two points i and j is smaller than the preset value of d_c. Furthermore, ρ_i is the number of points whose distances from point i are less than d_c. In Equation (2), δ_i is measured by calculating the minimum distance between point i and any other point with higher density. However, if the point i has the highest density, its δ value cannot be calculated by Equation (2). So, for the point i with the highest density, δ_i is calculated by Equation (3).

$$\delta_i = \max_j \left(d_{ij}\right) \tag{3}$$

SNN-DPC was proposed in the literature [33]. In SNN-DPC, the δ_i of the point i with the highest density is defined by Equation (4). Although Equation (3) and Equation (4) are different, these two calculating methods all can confirm central points of clusters.

$$\delta_i = \max_{i \neq j} \left(\delta_j\right) \tag{4}$$

For the Jain data set in which two crescent-shaped clusters of different densities are intertwined with each other, SNN-DPC can cluster the data sets of this type perfectly, but the clustering effect of DPC is not perfect. For the Path-based data set, the clustering effect of SNN-DPC is also better than DPC.

The time complexity and space complexity of DPC are both $O(m^2)$. Some studies in the literature [33–35] proposed the improved methods based on DPC, but the time complexity and space complexity limit these algorithms to be performed in sensor nodes. If the data set is small, the choice of d_c impacts on the clustering result greatly.

DBSCAN [36] does have some drawbacks, as the literature [33] points out. The main one relates to the declining clustering quality of high-dimensional data and variable-density clusters. However, it does not matter in our scenarios, where the dimension of data is only two, and the difference in the clusters density is less. As for the difficulty in setting parameters—such as Eps and Minpt–adaptive methods are employed in many recent papers, just as in our DCA-SF. Moreover, it is the permissive conditions and strong anti-noise capability [33] that make DBSCAN a proper solution in our scenarios. The DCA-SF that is proposed in the paper is a light detection scheme against selective forwarding attacks. The DCA-SF based on the DP-DBSCAN utilizes the clustering points of CFRs to detect attacks, which is different from E-DBSCAN. It achieves a low FDR and MDR with a low extra energy consumption of nodes.

3. Details of Scheme

DCA-SF assumes that the most dangerous attacks are from CHs on a WSN. This section will describe the scheme's details including the network layout, the cooperation of different roles of nodes, the density-based DCA, and the implementation of DCA-SF.

3.1. Frame of Detection Mechanism

The frame of detection is shown in Figure 3. One cluster contains three types of nodes: the cluster head (CH), member nodes (MNs), and the inspector node (IN). The cluster radius is half of the node communication radius. This ensures that the CHs and INs of adjacent clusters are within each other's communication range.

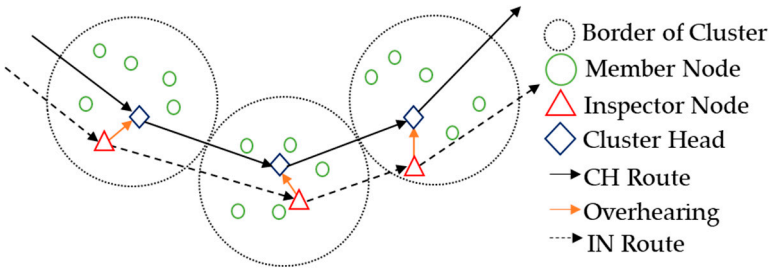

Figure 3. Detection frame.

MNs in one cluster send data packets consisting of information of the environment to the CH in the cluster. Then, the CH passes data packets to the next-hop CH along the CH route in Figure 3, until the data reach the SN. The IN calculates the CFR of the CH and CFRs of MNs in its cluster (the calculation method will be given in Section 3.2), and then passes them along the IN route to the SN shown in Figure 3, with the terminal of SN. Once the clustering of the network finished, the nodes in each cluster are determined, and the MNs in each cluster take turns as CHs and INs according to the detection results and election rules. In the i-th cluster, the MN with the highest residual energy will become a new CH_i and the MN with the second-highest residual energy will become a new IN_i.

3.2. Cooperation of Different Roles of Nodes

Four roles of nodes, MN, CH, IN, and SN exist in the WSN. As mentioned, MNs play the roles of the CH and IN in turn. Most nodes are MNs, and what they are supposed to do in the cluster-based WSN is merely to collect the information of the environment and send these data packets to their CHs. Next, CHs transfer the data packets they received to the SN. (Assume that there is no data fusion at CHs.)

INs do not forward any data packets during this process. Instead, IN_i calculates the CFR of CH_i (CFR_CH_i) and the CFR of MNs (CFR_MN_j) in the i-th cluster. Equations (5)–(7) and Equations (8) and (9) tell the details.

$$RECI\_CH_i(n) = \begin{cases} RECI\_CH_i(n-1) + reci\_CH_i(n), & n > 1 \\ reci\_CH_i(n), & n = 1 \end{cases} \tag{5}$$

$$FORW\_CH_i(n) = \begin{cases} FORW\_CH_i(n-1) + forw\_CH_i(n), & n > 1 \\ forw\_CH_i(n), & n = 1 \end{cases} \tag{6}$$

$$CFR\_CH_i(n) = \frac{FORW\_CH_i(n)}{RECI\_CH_i(n)} \tag{7}$$

$RECI\_CH_i(n)$ and $FORW\_CH_i(n)$ are the total numbers of packets received by CH_i and that of packets successfully forwarded by CH_i in the first n rounds, while the lowercase letters refer to the current round or the n-th round.

$$FORW\_MN_j(n) = \begin{cases} FORW\_MN_j(n-1) + forw\_MN_j(n), & n > 1 \\ forw\_MN_j(n), & n = 1 \end{cases} \tag{8}$$

$$CFR\_MN_j(n) = \frac{FORW\_MN_j(n)}{n} \tag{9}$$

in data-gathering WSNs, every MN sends one data packet to its CHs in each round. Obviously, the MN_j totally sends n data packets, of which $FORW\_MN_j(n)$ of them reach the CH.

Only SNs and INs detect attacks, and MNs do not. Firstly, the SN screens the suspected CHs according to the CFR_CHs from the IN routes shown in Figure 3. The CHs charged as malicious may be innocent, and the misdetection is due to the poor channel at some local areas. Secondly, INs compare the CFRs of suspected CHs to those of their MNs, which are in the same channel condition with suspected CHs. Once the suspected CH keeps the same channel condition as its MNs in terms of the CFR, it will be released or it will be isolated.

3.3. DP-DBSCAN DCA

3.3.1. Introduction of DBSCAN

DCA is a procedure of gathering data satisfying some criteria into one group. This technology has already been widely applied in various fields including pattern recognition and machine learning [36]. In DCA-SF, DCA separates the CFRs of the malicious CHs from those of normal nodes.

We improve the DCA, which depends on DBSCAN. Eps and Minpts are two important parameters in DBSCAN. Eps is the radius of a point's neighborhood. When Eps is set, Minpts, the number of points, works as the density threshold. The following are some concepts and principles of the DBSCAN [36–39] with parameters (Eps, Minpts).

- In DCA-SF, the data set (DS) consists of points in two-dimensional space, i.e., $DS = \{x_1, x_2, \ldots, x_m\}$ where m is the number of CHs. When an SN receives the CFRs of CHs from INs in the n-th round, the value of x_i can be arranged by Equation (10). The calculation of CH_is CFR in the n-th round is given in Section 3.2. Setting x_i in this way can enhance the stability of data clustering results in successive rounds.

$$x_i = (CFR\_CH_i(n), CFR\_CH_i(n-1)) \tag{10}$$

- Eps-neighborhood of a point: The Eps-neighborhood of a point x_i is defined by

$$N_{Eps}(x_i) = \left\{x_j \in DS \middle| dist\left(x_i, x_j\right) \le Eps\right\} \tag{11}$$

  where $dist(x_i, x_j)$ is the distance between x_i and x_j, and $N_{Eps}(x_i)$ is the Eps-neighborhood of x_i.
- Directly density-reachable: A point x_j is density-reachable from a point x_i if (1) $x_j \in N_{Eps}(x_i)$ and (2) $N_{Eps}(x_i) \ge Minpts$. If a point satisfies condition (2), this point is a core point.
- Density-reachable: If $\exists$ a chain of points $x_1, x_2, \ldots, x_c \in DS$, where any two successive points are directly density-reachable, then the points x_1 and x_c are density-reachable. If a point's density is not a core point but it is density-reachable from a core point, then this point is a border point.
- Density-connected: A point x_j is density-connected to a point x_i if there is a point $y \in D$, where both x_j and x_i are density-reachable from y.
- Data cluster: A data cluster (DC) is a nonempty subset of DS satisfying:

  (1) $\forall$ x_i,x_j, if $x_i \in DC$ and x_j is density-reachable from x_i, then $x_j \in DC$.
  (2) $\forall$ x_i,$x_j \in DC$, x_i is density-connected to x_j.

- Noise: Let $DC_1, DC_2, \ldots, DC_k$ be data clusters of the data set DS. Noise is the point not belonging to any data cluster DC_i in the data set DS, i.e., $noise = \{p \in DS | \forall\ i:\ p \notin DC_i\}$ $i = 1, 2, \ldots, k$.

DBSCAN divides data points into different data clusters based on density. The judgment is whether the number of points in the circle Eps-neighborhood is larger than or equal to Minpts. If there are at least Minpts points inside a point's Eps-neighborhood, the point is called a core point.

If there are less than Minpts points inside a point's Eps-neighborhood, and this point is inside a core point's Eps-neighborhood, then it is called a border point. Other points are called noise points. The pseudo-code of the algorithm is given below.

```
Input: sample data set DS, parameters (Eps, Minpts)
Output: data cluster set C
DBSCAN (DS, Eps, Minpts)
Begin
1.  Mark all points in DS as unvisited;
2.  Do
3.  Randomly choose an unvisited x_i;
4.  Mark x_i as visited;
5.  If points in x_i's Eps-neighborhood are no less than Minpts
6.    Create a new data cluster (DC);
7.    Set N consist of points in x_i's Eps-neighborhood;
8.      For each point x_j in N
9.        If x_j is unvisited
10.         Mark x_j as visited;
11.         If points in x_j's Eps-neighborhood are no less than Minpts, add points to N;
12.         End if
13.         If x_j is not a member of any data cluster, add x_j to DC;
14.         End if
15.       End if
16.     End for;
17.     Output DC;
18. Else mark x_i as a noise point;
19. End if
20. Until all unvisited points are visited;
21. Output C
End
```

3.3.2. Dynamic Parameter DBSCAN (DP-DBSCAN)

In our DCA-SF, the parameters (Eps, Minpts) change depending on the network scenarios. It is the dynamic parameters instead of the preset ones based on previous experience [36] that DP-DBSCAN employs to offer better reliability in changing situations. The methods of regulating these parameters adaptively in [40] are so complicated that they cannot be applied in a WSN. To apply the DP-DBSCAN to WSNs, we design a method to set Eps and Minpts, expecting a better performance in detecting a selectively forwarding attack.

The value of Eps is determined by the distribution of points in the data set. If the Eps is too large, a noise point would be judged as a normal one; if the Eps is too small, a normal point would be judged as a noise point. Based on the two-dimensional data set including $d_1 = (x_1, y_1)$, $d_2 = (x_2, y_2)$, ... , $d_m = (x_m, y_m)$, the center point $d = (x', y')$ is calculated by Equations (12) and (13). Then, the Eps is calculated by Equation (14), where $distance(p, q)$ denotes the Euclidean distance between points p and q.

$$x' = \frac{1}{m} * \sum_{i=1}^{m} x_i \tag{12}$$

$$y' = \frac{1}{m} * \sum_{i=1}^{m} y_i \tag{13}$$

$$Eps = \frac{1}{m} * (distance(d_1, d) + distance(d_2, d) + \ldots + distance(d_m, d)) \tag{14}$$

Minpts is especially defined in Equation (15).

$$Minpts = (c/b) + 0.5 \tag{15}$$

where c is the number of clusters in the WSN and b is an integer $1 < b < 12$, which will be discussed in Section 4.2.2.

3.3.3. Complexity Analysis of DP-DBSCAN

The time complexity of ordinary DBSCAN is O(m^2) [41], where m is the number of samples in the data set that needs to be clustered. The work of [41–43] reduces the time complexity to O(m) or O(m*log (m)), respectively. In DCA-SF, the DBSCAN algorithm is not too complex for SN and INs.

To calculate the parameters (Eps, Minpts) for DBSCAN, 2(m–1) + m additions and three divisions are performed in Equations (12)–(14). In addition, m times distance calculations need to be performed. So, the time complexity of DP-DBSCAN is O(5m) = O(m). In the proposed scheme, m is the number of CHs when it performs in the SN or sensor nodes in one WSN cluster when it performs in INs.

3.4. Implementation of DCA-SF

3.4.1. Two Key Notes

One key point in DCA-SF is that the CFRs of nodes are employed. Tables 1 and 2 give an intuitive grasp of the advantage. In a WSN, because the CFR is more stable than the SFR (single round forwarding rate), DCA-SF can pick malicious CHs out by their CFRs.

Table 1. Cumulative forwarding rates (CFR) of a cluster head (CH).

Round	1	2	3	4	5	6	7	8	9	10
CFR	0.9091	0.9545	0.9394	0.9318	0.9444	0.9375	0.9333	0.9186	0.8958	0.8962

Table 2. Single forwarding rate (FR) of a CH.

Round	1	2	3	4	5	6	7	8	9	10
Single FR	0.9231	1.0000	0.8463	0.7692	0.5385	1.0000	0.9231	0.9186	0.8463	0.9186

The other key point in DCA-SF is that the SN charges any CH as a suspect only when this CH is detected as abnormal in the successive k rounds. Now, the detection enters the called stable status. The earlier the stable status occurs, the faster the detection. After that, the suspected CH will be judged as malicious or innocent by the SN according to the report from its IN.

3.4.2. Process of DCA-SF

The DCA-SF scheme consists of centralized and distributed schemes. In the centralized detection scheme, the SN executes DP-DBSCAN independently after receiving CFRs of all the CHs from INs. Conducting centralized testing takes place for two reasons. On one hand, the SN is strong enough to implement the DP-DBSCAN without caring for the algorithm's time complexity and space complexity. On the other hand, the SN can never become malicious as assumed at the beginning of Section 3; thus, the results of clustering CH CFRs by the SN are highly reliable. If the CFR of a CH is determined as a noise by the SN, it is anomalous to other CHs' CFRs. Only if the CFR_CH_i is judged as noise in k

successive rounds can CH_i be marked as a suspect node. The result may come from a malicious node or a poor channel. IN_i performs DP-DBSCAN on the CFRs of CH_i and MNs in cluster i to confirm whether the CH_i is malicious or not. This processing is called distributed detection.

If k is too small, the MDR will be low. However, it will increase the number of times one CH is marked as suspect, which will increase the number of times that IN performs detection. The computational energy consumption of IN increases. If k is too large, the FDR will be low. However, a malicious CH that should be marked as the suspect may exist in the network for a long time, increasing the number of dropped packets. The value of k is set in advance and is fixed during network running. So, the value of k depends on the user's needs for the detection results. If the user requires a lower MDR, the value of k can be set smaller. If the user requires a lower FDR, the value of k can be set larger. In our simulation, the value of k is set to 4, 5 and 6. Figure 4 shows the detection process of the proposal scheme DCA-SF, where IN_i and CH_i represent the IN and CH of cluster i, respectively.

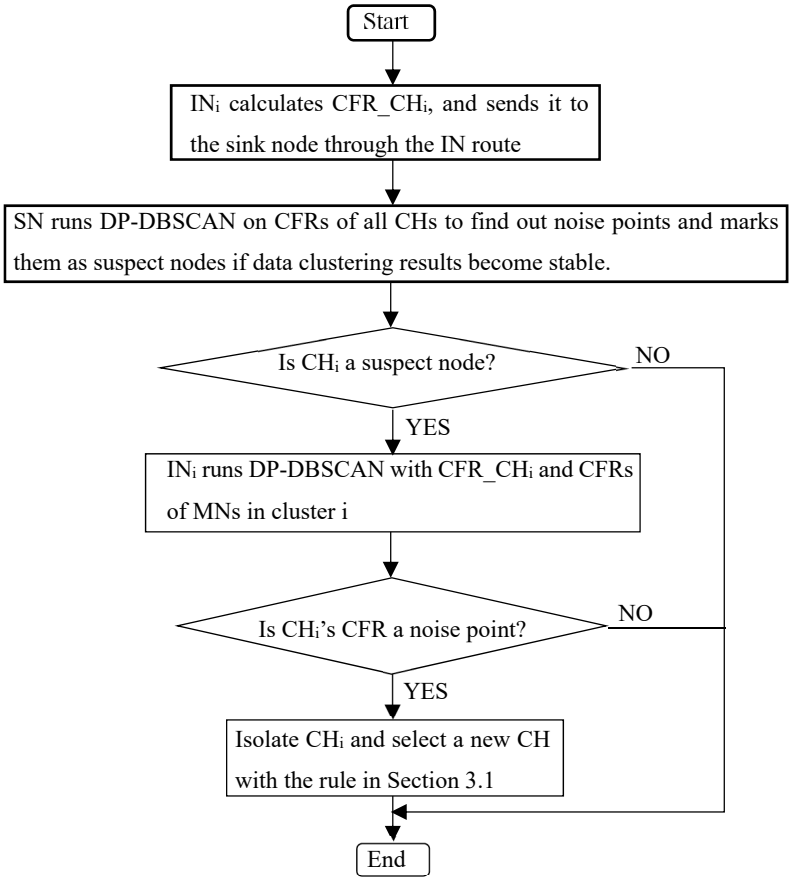

Figure 4. Detection flowchart.

4. Simulation Results and Analysis

4.1. Simulation Parameters and Data Set

4.1.1. Simulation Parameters

The simulation parameters in MATLAB 2014a are shown in Table 3.

The function rand (1, 1) in Table 3 is to generate numbers subjected to randomly uniform in (0, 1). The LEACH clustering method is used in our simulations; the selecting of CH and the forming of the cluster are random. In simulations, we define two metrics (MDR, FDR) as follows. The missed detection means that a malicious node is regarded as normal; the missed detection rate (MDR) is the ratio of missed detection CHs to the total malicious CHs. The false detection means that a normal

node is regarded as malicious; the false detection rate (FDR) is the ratio of false detection CHs to the total CHs.

According to the results of DP-DBSCAN, the point will be judged as noise if it is clustered as class 0.

Table 3. Parameters of simulation. MN: member node, WSN: wireless sensor network.

Parameter Item	Parameter
Clustering method of WSN	LEACH
Area of the network	300 × 300
Total number of nodes	300
Ratio of malicious nodes	0.05 + 0.2 × rand(1, 1)
Total forwarding rate of normal nodes	[0.8, 1] (Change as channel quality changes)
Total forwarding rate of malicious node	0.7 × rand(1, 1)
Total forwarding rate of MN	[0.8, 1] (Change as channel quality changes)
The number and rounds of dropping packets	Random

4.1.2. Data Set

As shown in Table 3, in simulation, the number of malicious CHs is the total number of CHs x the ratio of malicious CHs. It is subjected to a random uniform distribution. Then, an id set (IS) of malicious CHs is formed randomly. If the id of a CH belongs to IS, this CH will be malicious. We assume the total forwarding rates of malicious CHs (MNs) and normal CHs (MNs) obey a random uniform distribution in [0, 0.7] and [0.8, 1], respectively. The latter depends on channel quality. The dropped packets by CHs can be counted out. So, the number and rounds of malicious dropping packets are random, and the data set will be formed in every round by calculating the CFRs of CHs and MNs with Equations (7) and (9).

4.2. Simulation Results and Analysis

4.2.1. An Example of Simulation

In our simulations, the CHs' distribution according to the LEACH algorithm is identical to that in Figure 5. As shown in Figure 5, the network has been divided into 15 clusters in this simulation, and each cluster has selected a CH and an IN.

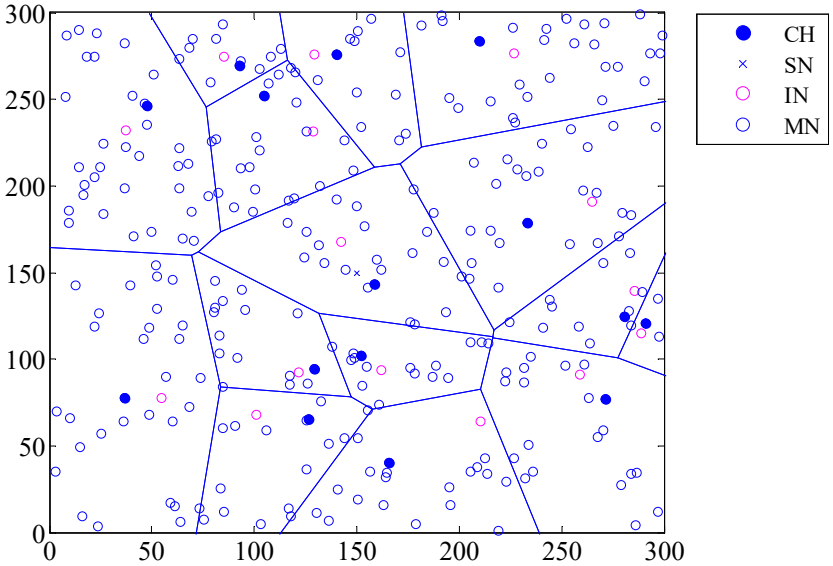

Figure 5. Network nodes distribution.

The total forwarding rates produce randomly as listed in Table 4. In this simulation, the total forwarding rates of normal CHs are located at [0.8052, 0.9402]. CH_2, CH_8, and CH_{12} are malicious nodes. The total forwarding rates of these three malicious nodes are 0.6299, 0.1468, and 0.6051 respectively, and assume that k described in Section 3.4.2 is set to five in this simulation.

Table 4. Total forwarding rate of each CH.

No. CH	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Forwarding rate	0.8755	0.6299	0.9074	0.8057	0.8052	0.9402	0.9030	0.1468
No. CH	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
Forwarding rate	0.8256	0.8851	0.9336	0.6051	0.9017	0.8862	0.8183	

The classification of each CH after DP-DBSCAN clustering in each round is shown in Table 5. The CFR of CH_8 is clustered as zero classifications from the first round. This shows that CH_8 has dropped many packets from the first round. Different from CH_8, the total forwarding rates of CH_2 and CH_{12} are closer to normal; their CFRs are clustered as class 1 in the first five rounds. However, as the network runs, their CFRs gradually show differences. So, they are clustered as zero after the fifth round. The detection results become stable after the fifth round. The SN broadcasts the message (CH_2, CH_8, and CH_{12} are regarded as suspected) to IN_2, IN_8, and IN_{12} respectively. After IN_2, IN_8, and IN_{12} have done DP-DBSCAN in their clusters, they are confirmed as malicious. So, the MDR and FDR in this simulation are both zero.

Table 5. CFRs clustering results in each round.

No. CH \ **No. Round**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	⋯	**49**	**50**
1	1	1	1	1	1	1	1	1	1	1	…	1	1
2	1	1	1	1	1	0	0	0	0	0	…	0	0
3	1	1	1	1	1	1	1	1	1	1	…	1	1
4	1	1	1	1	1	1	1	1	1	1	…	1	1
5	1	1	1	1	1	1	1	1	1	1	…	1	1
6	1	1	1	1	1	1	1	1	1	1	…	1	1
7	1	1	1	1	1	1	1	1	1	1	…	1	1
8	0	0	0	0	0	0	0	0	0	0	…	0	0
9	1	1	1	1	1	1	1	1	1	1	…	1	1
10	1	1	1	1	1	1	1	1	1	1	…	1	1
11	1	1	1	1	1	1	1	1	1	1	…	1	1
12	1	1	1	1	1	0	0	0	0	0	…	0	0
13	1	1	1	1	1	1	1	1	1	1	…	1	1
14	1	1	1	1	1	1	1	1	1	1	…	1	1
15	1	1	1	1	1	1	1	1	1	1	…	1	1

4.2.2. Results and Analysis

To make it clear how Minpts affects the detection result, we perform 500 simulations and get the statistical results. In the 500 simulations, the ratios of malicious node are Ar = 5%, Ar = 10%, Ar = 15%, Ar = 20%, and Ar = 25%. In these 500 simulations, k described in Section 3.4.2 is set to five. In every ratio, we change the value of b in Equation (15) to conduct 100 simulations. Then, we get the relationships between MDR and b, FDR and b, and the number of rounds at which the detection results

becoming stable and b. Figure 6 shows the relationship between MDR and b in different malicious node ratios.

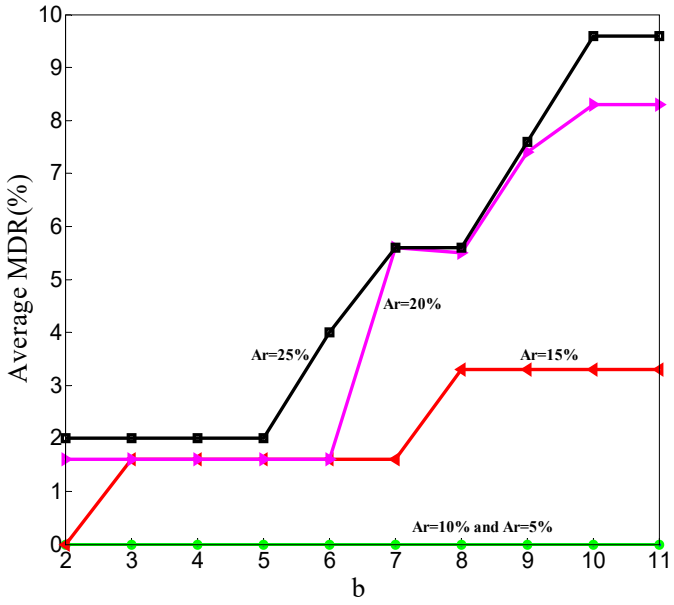

Figure 6. Relationship between average missed detection rate (MDR) and b.

In Figure 6, the curve representing Ar = 5% is covered by the curve representing Ar = 10%. The MDR is 0 in both situations where Ar = 10% and Ar = 5%. Minpts changes with the number of CHs in WSNs. The average MDR under different malicious nodes is counted out by 10 simulations in the same ratio and b. The overall trend is that the average MDR increases as the b value increases.

In Figure 7, the curve representing Ar = 20% is covered by the curve representing Ar = 25%. FDR is 0 in the both situations of Ar = 20% and Ar = 25%. The relationship between the average FDR and b is shown in Figure 7, in which the average FDR reduces as b increases.

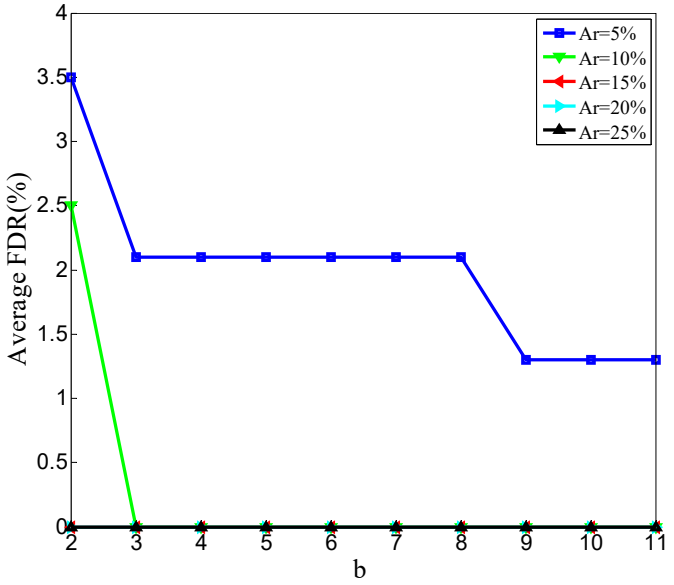

Figure 7. Relationship between average false detection rate (FDR) and b.

The relationship between average stable rounds and b is shown in Figure 8.

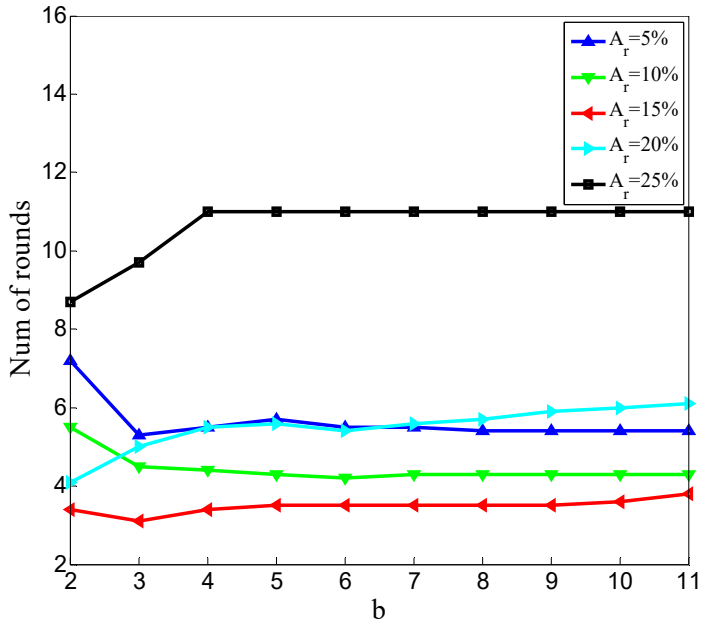

Figure 8. Relationship between the average stable rounds at which the detection results begin to stabilize and b.

The average rounds shown in Figure 8 are counted to reflect the detection speed. The fewer rounds required for the detection results to stabilize, the higher the detecting speed. The overall trend shown in Figure 8 is not obvious, and an obvious overall trend will be shown in the following.

In Figure 9, the overall trend is obviously shown. The average MDR increases as the b value increases. The average MDR of $k = 6$ is higher than that of $k = 5$, and so $k = 5$ is set to $k = 4$.

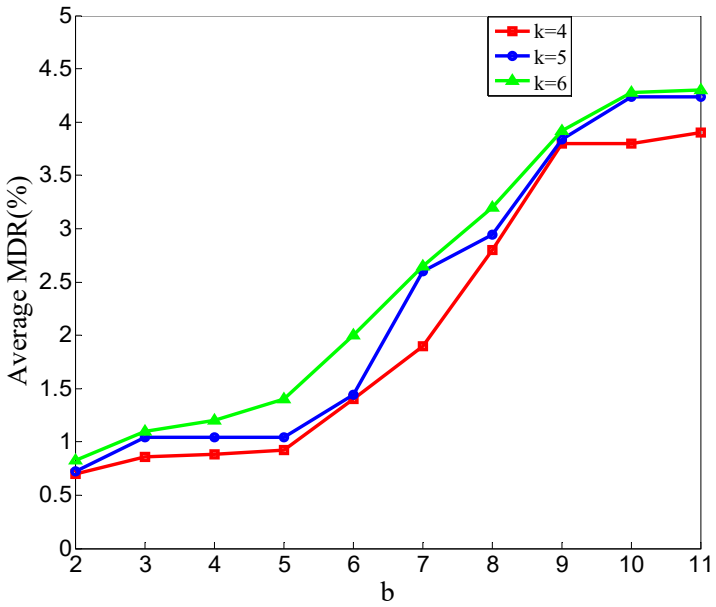

Figure 9. Average missed detection rate (MDR) of all experiments in different values of b.

In Figure 10, the average FDR decreases as the b value increases; the average FDR of $k = 4$ is higher than that of $k = 5$, and that of $k = 5$ is higher than that of $k = 6$.

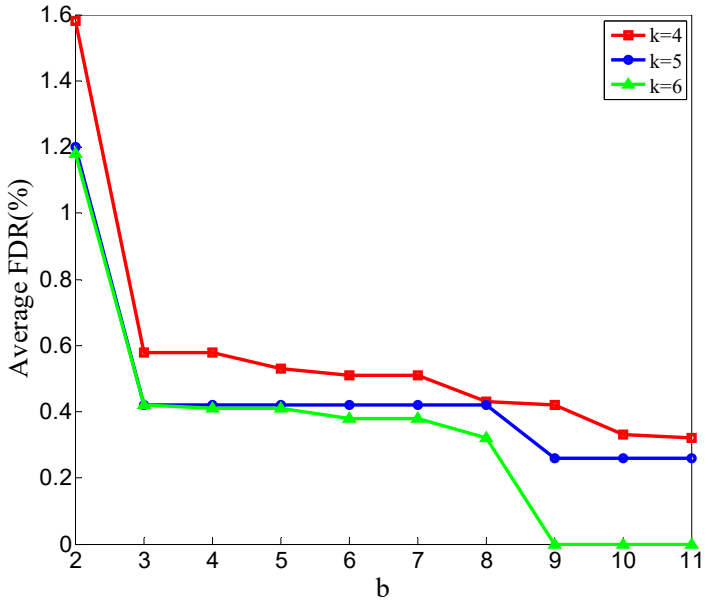

Figure 10. Average FDR of all experiments in different values of b.

Figure 11 shows the average number of rounds at which the detection results become stable for all the simulations for different values of b in situations of $k = 4$, $k = 5$, and $k = 6$.

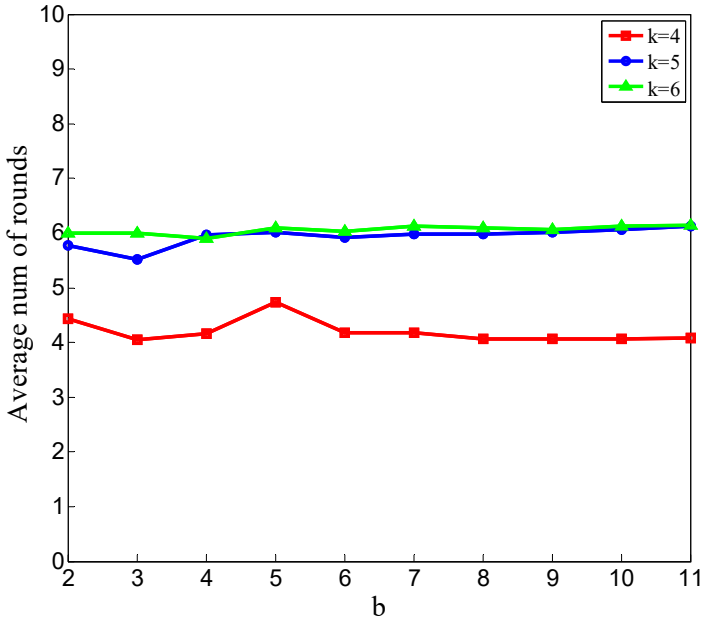

Figure 11. Average stable rounds at which the detection results begin to stabilize for different values of b.

According to the statistical results, when b = 3 and $k = 5$, the average MDR of DCA-SF is 1.04%, the average FDR of DCA-SF is 0.42%, and the detection results become stable from an average of 5.5 rounds.

Furthermore, the value of b may change as the application scenario changes. If users focus more on the lower MDR, the value of b can be taken as smaller; if users focus more on the lower FDR, the value of b can be taken as larger.

4.2.3. Results Analysis

As shown in Figures 6 and 9, the average MDR increases as the b value increases. This conclusion means that when Eps and c are constant, the average MDR increases as the Minpts decreases. This is because when the neighborhood radius is fixed, decreasing the value of Minpts will make the point with fewer points in its neighborhood not be judged as noise. All points will not be judged as noisy if Minpts reduces to zero. This will lead our scheme to mark all the CFRs of malicious nodes as normal. With the same reason, all the CFRs of normal nodes are not marked as anomalous. So, the average FDR reduces as the Minpts decreases; the results are shown in Figures 7 and 10.

As shown in Figure 11, the average rounds at which the detection results become stable is almost stable as b changes because this average number is determined by the manner of dropping packets. In our simulation, the number and rounds of malicious nodes dropping packets are random, so the time at which the detection results begin to stabilize is little correlated to Minpts.

The results show that small k values usually make MDRs better while large k values bring low FDRs. Users can set k according to their requirements.

4.3. Comparison to Other Schemes

4.3.1. Detection Results Compared with Other DCAs

In this section, we compare the clustering results of DP-DBSCAN in DCA-SF with those of K-means in terms of the MDR and FDR. When K = 2, most of the CFRs in one cluster are normal, while the CFRs in the other cluster are anomalous. When K = 3, three clusters are normal, anomalous, and suspected, respectively. A total of 150 simulations are performed to compare the performance in different DCAs; in these 150 simulations, we set b = 3, k = 5.

As shown in Figure 12, in terms of the MDR, the performance of DP-DBSCAN is better than K-means (K = 2) and K-means (K = 3), and the time complexities of these three DCAs are all O(m).

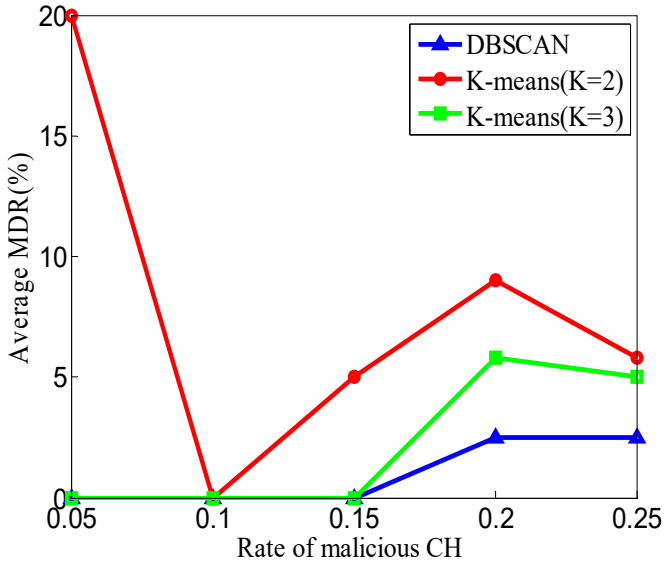

Figure 12. Comparison of MDR in different data clustering algorithms (DCAs).

The comparison of FDRs in different DCAs is shown in Figure 13. The results of the FDR in DP-DBSCAN are the same as those in K-means when K = 2, and the performances of them are better than those when K = 3.

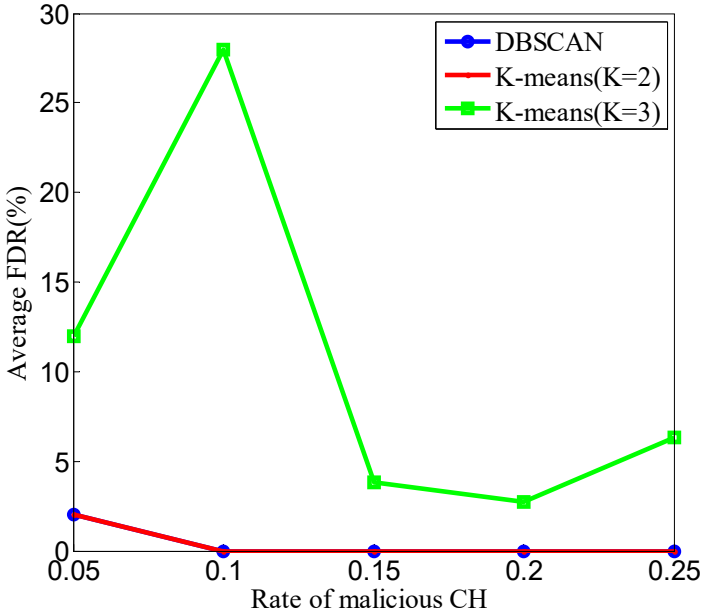

Figure 13. Comparison of FDR in different DCAs.

As shown in Figures 12 and 13, at the same time as the complexity O(m), the detection results of DP-DBSCAN are the best in these three DCAs. If the *K* value of K-means can be determined by some optimization algorithms, it can get a better result, but the time complexity and space complexity are too high to be performed in sensor nodes.

4.3.2. Detection Results Compared with Other Schemes

We compare DCA-SF with the watchdog mechanism [15], neighbor-based scheme [13], and IN-based scheme [8] on two metrics of MDR and FDR. The behaviors of these four schemes are recorded in Table 6. The simulations on all four schemes are on the same scenarios.

Table 6. Comparison of four schemes.

Scheme \ Metric	Ratio of Malicious Node	Number of Simulations	MDR	FDR
Watchdog [15]	10%	100	4%	28.2%
Neighbor-based monitoring [13]	10%	100	3%	17.2%
IN monitoring [8]	10%	100	1%	2.5%
DCA-SF	10%	100	0	0

4.3.3. Energy Consumption Compared with Other Schemes

All nodes take part in detecting selective forwarding attacks directly in the watchdog mechanism [15] and the neighbor-based scheme [13], despite the cost of energy. In other words, these two schemes are far from energy-efficienct. So, we only compare our scheme with IN monitoring [8] in the metric of energy consumption.

Figure 14 shows an ideal deployment of sensor nodes in a WSN. More nodes lay near the SN (at the center) because they have to forward more data packets from nodes far from the SN in the multi-hop WSNs.

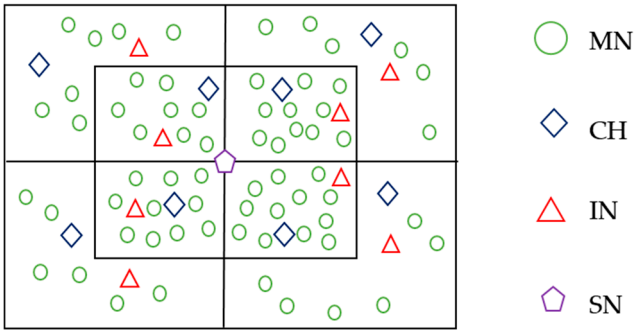

Figure 14. The best distribution of nodes in view of the energy balance.

Figure 15 is the simulation result with the energy consumption module [44] in wireless communication.

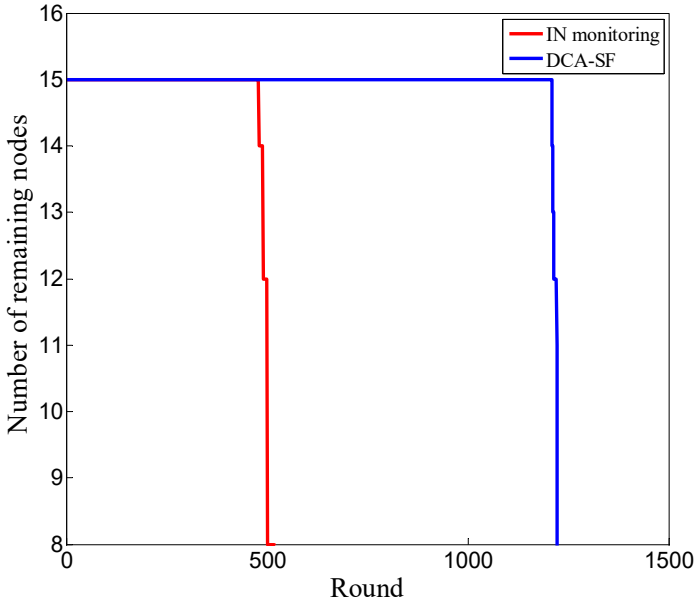

Figure 15. Comparison of network lifetime.

DCA-SF offers the WSN a much longer lifetime than the IN monitoring scheme. Under the same simulation settings, the death of node occurs firstly in the 1210th round with DCA-SF, which is nearly 700 rounds later than when it occurs with IN monitoring. The fewer nodes involved, the less energy consumed in detection. Nodes taking turns as CHs and INs also helps balance the energy over the WSN, which contributes to the long lifetime of the network.

5. Conclusion and Future Work

To detect selective forwarding attacks in WSNs, we have proposed the DCA-SF scheme based on the DBSCAN. The DP-DBSCAN, based on DBSCAN, can cluster the abnormal behaviors of malicious nodes to improve the scheme's intelligence. One is the radius of neighborhood Eps in DCA-SF determined by the distribution of points in the data set, and the other is Minpts in DP-DBSCAN, which is dependent on the number of points in the data set. When b = 3, k = 5, the statistical results show that the DCA-SF has an MDR of 1.04% and an FDR of 0.42%. The detection results become stable after an average of 5.5 rounds.

In the future, we will take a field test of this detection scheme on our WSN platform. Then, we will apply DP-DBSCAN to distributed WSNs. We plan to employ DPC and SNN-DPC to cluster the data set in SN for testing their effects.

Author Contributions: This paper was completed from the cooperative efforts of all authors. Conceptualization, H.F.; Methodology, H.F. and Y.L.; Validation, H.F., Y.L.; Formal analysis, H.F., Y.L., and Z.D.; Investigation, H.F., Z.D.; Resource, Y.W.; Data processing, H.F.; Writing—Original draft preparation, H.F.; Writing—Review and editing, Y.W., Y.L., Z.D., and H.F.; Supervision, Y.W.; Project administration, Y.W.; Funding acquisition, Z.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by CERNET Innovation Project grant number (NGII 20180317) and CERNET Innovation Project grant number (NGII 20180317).

Acknowledgments: This research is supported by the CERNET Innovation Project (NGII 20180317).

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

References

1. Zhou, H.B.; Wu, Y.M.; Hu, Y.Q. A Novel Stable Selection and Reliable Transmission Protocol for Clustered Heterogeneous Wireless Sensor Networks. *Comput. Commun.* **2010**, *33*, 1843–1849. [CrossRef]
2. Liu, Y.H.; Wu, Y.M.; Chang, J.Y. The Diffusion Clustering Scheme and Hybrid Energy Balanced Routing Protocol (DCRP) in Multi-hop Wireless Sensor Networks. *Ad Hoc Sens. Wirel. Netw.* **2019**, *43*, 33–56.
3. Wu, Y.M. An Energy-balanced Loop-free Routing Protocol for Distributed Wireless Sensor Networks. *Int. J. Sens. Netw.* **2017**, *23*, 123–131. [CrossRef]
4. Liu, Y.H.; Wu, Y.M. A Key Pre-distribution Scheme based on Sub-regions for Multi-Hop Wireless Sensor Networks. *Wirel. Pers. Commun.* **2019**, *109*, 1161–1180. [CrossRef]
5. Gulhane, G.; Mahajan, N. Performance Evaluation of Wireless Sensor Network under Black Hole Attack. *Int. J. Comput. Technol.* **2014**, *1*, 92–96.
6. Lu, Z.; Sagduyu, Y.; Li, J. Queuing the trust: Secure Backpressure Algorithm against Insider Threats in Wireless Networks. In Proceedings of the IEEE Conference on Computer Communications (INFOCOM' 2015), Hong Kong, China, 26 April–1 May 2015.
7. Chae, Y.; Dipippo, L.C. Trust Management for Defending On-Off Attacks. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 1178–1191. [CrossRef]
8. Zhou, H.; Wu, Y.M.; Feng, L. A Security Mechanism for Cluster-Based WSN against Selective Forwarding. *Sensors* **2016**, *9*, 1537. [CrossRef]
9. Butun, I.; Morgera, S.D.; Sankar, R. A Survey of Intrusion Detection Systems in Wireless Sensor Networks. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 266–282. [CrossRef]
10. Chris, K.; David, W. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. In Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, Anchorage, AK, USA, 11 May 2003.
11. Semanti, D.; Abhijit, D. An Algorithm to Detect Malicious Nodes in Wireless Sensor Network Using Enhanced LEACH Protocol. In Proceedings of the 2015 International Conference on Advances in Computer Engineering and Applications, Ghaziabad, India, 19–20 March 2015.
12. Ishaq, Z.; Park, S.; Yoo, Y. A Security Framework for Cluster-Based Wireless Sensor Networks against the Selfishness Problem. *Wirel. Commun. Mob. Comput.* **2018**, *2018*, 11. [CrossRef]
13. Tseng, C.Y.; Balasubramanyam, P.; Ko, C.; Limprasittiporn, R.; Rowe, J.; Levitt, K. A Specification-Based Intrusion Detection System for AODV. In Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, Fairfax, VA, USA, 31 October 2003.
14. Otoum, S.; Kantarci, B.; Mouftah, H.T. Mitigating False Negative Intruder Decisions in WSN-based Smart Grid Monitoring. In Proceedings of the IEEE 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC'2017), Valencia, Spain, 26–30 June 2017.
15. Marti, S.; Giuli, T.J.; Lai, K.; Baker, M. Mitigating Routing Misbehavior in Mobile and Ad Hoc Networks. In Proceedings of the International Conference on Mobile Computing and Networking (Mobicom'2000), Boston, MA, USA, 6–11 August 2000.
16. Cho, Y.H.; Qu, G.; Wu, Y.M. Insider Threats against Trust Mechanism with Watchdog and Defending Approaches in Wireless Sensor Networks. In Proceedings of the 2012 IEEE Symposium on Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24–25 May 2012.

17. Nii, E.; Kitanouma, T.; Adachi, N.; Takizawa, Y. Cooperative Detection for Falsification and Isolation of Malicious Nodes for Wireless Sensor Networks in Open Environment. In Proceedings of the 2017 IEEE Asia Pacific Microwave Conference (APMC 2017), Kuala Lumpur, Malaysia, 13–16 November 2017.
18. Ma, D.Y.; Zhang, A.D. An Adaptive Density-based Clustering Algorithm for Spatial Database with Noise. In Proceedings of the 4th IEEE International Conference on Data Mining (ICDM'2004), Brighton, UK, 1–4 November 2004.
19. Zhang, J.; Zulkernine, M.; Haque, A. Random-forests-based Network Intrusion Detection Systems. *IEEE Trans. Syst. Man Cybern Part C* **2008**, *38*, 649–659. [CrossRef]
20. Kdd Cup 1999 Data. Available online: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 28 May 2019).
21. Guha, S.; Rastogi, R.; Shim, K. Cure: An efficient clustering algorithm for large databases. *Inf. Syst.* **2001**, *26*, 35–58. [CrossRef]
22. Zhang, T.; Ramakrishnan, R.; Livny, M. Birch: An efficient data clustering method for very large databases. In Proceedings of the ACM Sigmod Record, Montreal, QC, Canada, 4–6 June 1996.
23. Ulrike, V. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416.
24. Wang, W.; Yang, J.; Muntz, R. Sting: A statistical information grid approach to spatial data mining. In Proceedings of the VLDB, Athens, Greece, 25–29 August 1997.
25. MacQueen, J. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*; UC Press: Berkeley, CA, USA, 1967.
26. Park, H.; Jun, C. A Simple and Fast Algorithm for K-medoids Clustering. *Expert Syst. Appl.* **2009**, *36*, 3336–3341. [CrossRef]
27. Rodriguez, A.; Laio, A. Clustering by Fast Search and Find of Density Peaks. *Science* **2014**, *344*, 1492–1496. [CrossRef] [PubMed]
28. Xie, J.; Guo, W.; Xie, W.; Gao, X. Improved Rival Penalized Competitive Learning algorithm based on sample spatial distribution density. *J. Comput. Appl.* **2012**, *32*, 638–642.
29. Xu, L.; Krzyzak, A.; Oja, E. Rival Penalized Competitive Learning for Clustering Analysis, RFB Net, and Curve Detection. *IEEE Trans. Neural Netw.* **1993**, *4*, 636–649. [CrossRef]
30. Kaufman, L.; Rousseeuw, P.J. Clustering by means of medoids. In *Statistical Data Analysis Based on the L Norm*; Elsevier: Amsterdam, The Netherland, 1987; pp. 405–416.
31. Dhillon, I.; Guan, Y.; Kogan, J. Refining Clusters in High Dimensional Text data. In Proceedings of the 2nd SIAM Workshop on Clustering High Dimensional Data, Arlington, TX, USA, 13 April 2002.
32. Theodoridis, S.; Koutroumbas, K. *Pattern Recognition*, 4th ed.; Academic Press: Boston, MA, USA, 2009; pp. 745–748.
33. Liu, R.; Wang, H.; Yu, X. Shared-nearest-neighbor-based clustering by fast search and find of density peaks. *Inf. Sci.* **2014**, *450*, 200–226. [CrossRef]
34. Gong, S.; Zhang, Y. EDDPC: An Efficient Distributed Density Peaks Clustering Algorithm. *J. Comput.* **2016**, *53*, 1400–1409.
35. Zhang, Y.; Chenny, S.; Yu, G. Efficient Distributed Density Peaks for Clustering Large Data Sets in Mapreduce. In Proceedings of the IEEE International Conference on Data Engineering, San Diego, CA, USA, 19–22 April 2017.
36. Martin, E.; Hans-Peter, K.; Joerg, S.; Xu, X.W. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (ICKDDM' 1996), Warsaw, Portland, 2–4 August 1996.
37. Liu, P.; Zhou, D.; Wu, D. VDBSCAN: Varied Density Based Spatial Clustering of Applications with Noise. In Proceedings of the IEEE 4th International Conference on Service Systems and Service Management (ICSSSM' 2007), Chengdu, China, 9–11 June 2007.
38. Khan, K.; Rehman, S.U.; Aziz, K.; Fong, S.; Sarasvady, S. DBSCAN: Past, Present and Future. In Proceedings of the IEEE 4th International Conference on the Applications of Digital Information and Web Technologies(ICADIWT 2014), Chennai, India, 17–19 February 2014.
39. Berkin, P. *Survey of Clustering Data Mining Techniques. Grouping Multidimensional Data*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 43, pp. 25–71.
40. Karami, A.; Ronnie, J. Choosing DBSCAN Parameters Automatically Using Differential Evolution. *Int. J. Comput. Appl.* **2014**, *91*, 1–11. [CrossRef]

41. Borah, B.; Bhattacharyya, D.K. An Improved Sampling-based DBSCAN for Large Spatial Databases. In Proceedings of the IEEE International Conference on Intelligent Sensing and Information Processing (ICISIP'2004), Chennai, India, 4–7 January 2004.
42. Gan, J.H.; Tao, Y.F. DBSCAN Revisited: Mis-Claim, Un-Fixability, and Approximation. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD 2015), Melbourne, Australia, 31 May–4 June 2015.
43. Chen, Y.W.; Tang, S.Y.; Bouguila, N.; Wang, C.; Du, J.; Li, H.L. A Fast Clustering Algorithm Based on Pruning Unnecessary Distance Computations in DBSCAN for High-Dimensional Data. *Pattern Recognit.* **2018**, *83*, 375–387. [CrossRef]
44. Hu, Y.; Wu, Y.M.; Wang, H.S. Detection of Insider Selective Forwarding Attack Based on Monitor Node and Trust Mechanism in WSN. *Wirel. Sens. Netw.* **2014**, *6*, 237–248. [CrossRef]