*Ad Hoc & Sensor Wireless Networks*, Vol. 43, pp. 213–238
Reprints available directly from the publisher
Photocopying permitted by license only



# A New Algorithm to Defend Against Sybil Attack in Static Wireless Sensor Networks Using Mobile Observer Sensor Nodes

MOJTABA JAMSHIDI[1,*], MILAD RANJBARI[2], MEHDI ESNAASHARI[3], ASO MOHAMMAD DARWESH[4] AND MOHAMMAD REZA MEYBODI[5]

[1]*Department of Information Technology, University of Human Development, Sulaymaniyah, Iraq*
[2]*Department of Computer Engineering, Arak Branch, Islamic Azad University, Arak, Iran*
*E-mail: miladranjbari@gmail.com*
[3]*Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran*
*E-mail: esnaashari@kntu.ac.ir*
[4]*Department of Information Technology, University of Human Development, Sulaymaniyah, Iraq*
*E-mail: aso.darwesh@uhd.edu.iq*
[5]*Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran*
*E-mail: mmeybodi@aut.ac.ir*

*Received: December 16, 2016. Accepted: August 8, 2016.*

A well-known dangerous attacks against Wireless Sensor Networks (WSNs) is Sybil attack in which a malicious node propagates multiple fake identities. This attack affects routing protocols and many usual operations of WSNs including voting and data aggregation. In this paper, a novel algorithm has been proposed for detecting this attack in static WSNs. There are several mobile observer nodes in the proposed algorithm, which walk in the network's environment continuously. They first detect areas suspicious to Sybil attack and then record information about these areas in their memories. Using stored information, they will be able to detect Sybil nodes. The proposed algorithm is implemented using JSIM simulator and its efficiency has been compared to other existing algorithms in terms of true detection and false detection rates. The results show that the proposed algorithm can detect 99% of Sybil nodes on average, while the false detection rate is about 5%.

*Keywords:* Static wireless sensor network; security; Sybil attack, malicious node, observer node, JSIM simulator.

* Contact author: E-mail: jamshidi.mojtaba@gmail.com

## 1 INTRODUCTION

Wireless Sensor Networks are a type of ad-hoc network which provides the possibility for remote monitoring and controlling physical environments with better accuracy. Usually, there are hundreds and thousands of small and cheap sensor nodes in each sensor network. Considering sensor nodes' limitations in terms of energy, memory and computational power together with broadcast nature of wireless communication and being non-resistant against adversary tamper, establishing security in such networks is an important issue [1, 2].

One of the most important and effective attacks against network layer (routing) is the Sybil attack [3]. In Sybil attack, adversary either captures a normal node in the network and reprograms it (as a malicious node) or injects an illegal node to the network as a malicious node. This malicious node propagates multiple identities (which are called Sybil nodes hereafter) after deployment in the network, where these identities are fake which are either created by the adversary or stolen by the adversary from other normal nodes in other areas of the network. When the malicious node propagates multiple identities simultaneously, it attracts a heavy traffic because neighboring normal nodes assume each identity (Sybil node) belongs to an individual physical node while all these identities belong only to one hardware node. Therefore, this attack can disrupt routing protocols and many operations including voting, misbehavior detection, data aggregation, resource allocation, and reputation evaluation significantly [4, 5].

Several algorithms against Sybil attack in WSNs have already been proposed, where each algorithm employs a specific technique for detecting Sybil nodes or preventing the establishment of this attack. For example, [6] has proposed an algorithm for detecting Sybil node which is based on radio resource testing, in which each node assigns an individual channel to its neighbors for communication. Considering the limitations of sensor nodes, this approach cannot be efficient. In addition, [6] has proposed another method based on identity registration for detecting Sybil nodes which uses a trusted central authority and employs voting for identifying Sybil nodes. In general, protection mechanisms based on voting approach or reputation approach cannot perform efficiently, because some nodes are impostors and cannot be relied on for obtaining reliable information. Additionally, authentication methods such as [7] usually require a large amount of space for storing necessary authentication information (including shared encryption keys and identity certificates) and involve complicated monitoring algorithms. In addition, algorithms based on Received Signal Strength Indicator (RSSI) [8] cannot be employed for determining nodes' location and identifying Sybil nodes because radio signal is prone to be interfered by the environment, thus detection accuracy of such algorithms would be affected.

In this paper, a novel algorithm is proposed which uses mobile observer nodes for identifying Sybil nodes in static WSNs, such that the drawbacks of previous algorithms are resolved. The proposed algorithm does not need to locate nodes and only exploits information regarding neighborhood and nodes' density in different areas of the network for detecting Sybil attack.

The rest of this paper is organized as follows: Section 2 introduces the related works. In Section 3, system assumptions and attack model are discussed. Section 4 presents the proposed algorithm, while Section 5 presents the results of evaluation and simulation. The paper is concluded in Section 6.

## 2 RELATED WORK

In this section, we investigate the existing algorithms against Sybil attack in static and mobile WSNs.

### 2.1 Algorithms in Static WSNs

Sybil attack is introduced by Douceur [3] for peer-to-peer distributed systems. Karlof and Wagner [4] proved that the Sybil attack could disrupt routing protocols in WSNs. Newsome *et al.* [6] analyzed Sybil attack systematically and presented its taxonomy based on how Sybil identities are created and simultaneity of their indication (indicating them). They also proposed four methods to detect Sybil attacks, such as Radio Resource Test (RRT), Code Attestation (CA), Random Key Pre-distribution (RKP), Identity Registration (IR), and Position Verification (PV). In RRT method, each nodes assigns unique channels to its neighbours. This approach cannot be efficient and applicable considering sensor node limitations. The IR method uses voting and a central validation management unit in the network to detect illegal Sybil identities. Generally, the mechanisms based on voting or reputation approaches cannot detect Sybil attacks effectively because some nodes (Sybil nodes) are fake and obtaining reliable information from them is not reasonable.

Zhong *et al.* [8] proposed an RSSI-based algorithm that uses a few receiver nodes to estimate the location of each sender node. Demirbas and Song [9] have also used this mechanism by employing four detector nodes to estimate sensor nodes location in the network. This is enough for detecting Sybil nodes in the network because all Sybil nodes belong to malicious node have identical locations. Generally, the accuracy of RSSI-based algorithms is sharply dropped in noisy environments. An advanced RSSI-based algorithm was proposed by Mirsa and Myneni [10] which can detect the Sybil attack even if Sybil nodes change their transmission power for each Sybil identity. Also, two RSSI-based algorithms were proposed by Chen *et al.* [11] and Jamshidi

*et al.* [12] for detecting Sybil nodes in cluster-based sensor networks. The first algorithm can only detect Sybil nodes which appear as cluster heads and the second algorithm can only detect Sybil nodes which appear as cluster members.

Zhang *et al.* [13] proposed an Angle Of Arrival identification (AOA) based algorithm, named Trust Evaluation Algorithm Based on AOA (TEBA), which takes advantage of the fact that, although malicious nodes could create multi-identities, they all have the same physical position. In this algorithm there are some beacon nodes which identify Sybil identities with signal phase difference below a trusted threshold for adjacent sensor nodes. Saxena and Sejwar [14] proposed an algorithm for Sybil attack detection based on Time Difference Of Arrival (TDOA) localization method, which detects the malicious behavior of head nodes and member nodes in a cluster-based network.

Ssu *et al.* [15] proposed a distributed algorithm which uses the neighbouring information and network density to detect Sybil nodes. This algorithm is based on the assumption that the probability of two neighbour nodes having exactly the same set of neighbours is extremely low provided that the network has a high node density.

Rafeh and Khodadad [16] has proposed an algorithm which uses a two-hop message broadcasting to defend against Sybil attacks in static WSNs. This algorithm discovers the common neighbours between each couple of nodes which are two-hop neighbours by propagating messages with Time To Live (TTL=2). This algorithm marks the common neighbours as Sybil nodes if their number exceeds a threshold. Muraleedharam *et al.* [17] utilized a hybrid intelligence scheme to prevent the Sybil attack. This scheme uses the swarm intelligence algorithm to collect route's information. A Sybil node can be detected through its energy variation. The information collected by swarm agents is used as training data for the Bayesian network to adjust the threshold parameter.

A message authentication algorithm is proposed by Dhamodharan and Vayanaperumal [18] which is applied for detecting Sybil nodes. The node acts as a Sybil node with duplicated ID, and information can happen only when the node has complete information about other nodes. A Compare And Match-Position Verification Method (CAM-PVM) is used to verify the node's requirements. Also, a message authentication algorithm and a passing procedure are applied for authentication prior to communication. A node cannot communicate with any other nodes in the network, if it's not authorized by the base station.

A Random Password Generation (RPG) method is proposed by Amuthavalli and Bhuvaneswaran [19] to monitors the network traffic and security levels during data transmission. This RPG method provides the routing table which keeps information about the nodes in the network. This algorithm can detect Sybil nodes which appear as intermediate nodes between

source and destination nodes. In order to decide whether these intermediate nodes are Sybil or normal, the intermediate node's information is compared with the RPG database during communication.

A rule-based anomaly detection system is proposed by Sarigiannidis *et al.* [20], which relies on an Ultra-Wide Band (UWB) ranging-based detection method. Hu *et al.* [21] have proposed a one-way key chain ID authentication algorithm to decrease the probability for attackers to lunch Replica and Sybil attacks. They used elliptic curve discrete logarithm problem and node neighbour relationship to authorized nodes. Jan *et al.* [22] have proposed a two-tier detection algorithm to detect Sybil nodes in forest wildfire monitoring applications. In this algorithm, nodes with high-energy are responsible to detect Sybil nodes. In addition, two base stations are used to improve the detection rate of Sybil attacks.

### 2.2 Algorithms in Mobile WSNs

Jamshidi *et al.* [23] proposed an algorithm which uses observer nodes to detect Sybil nodes in mobile WSNs. This algorithm is based on the underlying idea of counting the number of sensor nodes facing with observer nodes, creating separate sets, and intersecting them. This algorithm consists of two phases, network traffic monitoring phase and the Sybil node detection phase. In the first phase, each observer node stores the occurrences of facing each other sensor nodes in its history, for $R$ rounds of network activities. The second phase consists of two steps. In the first step, each observer node $u$ generates distinct sets of node IDs, such that each set includes the IDs of nodes, which appeared for an equal number in its neighbourhood. Each observer node $u$ then selects the sets, whose members are larger or equal to a threshold $T_{min}$, as a suspicious Sybil list. In the second step, observer nodes cooperate to mark Sybil nodes. They send their suspicious Sybil list to each other. Each observer node creates a new list by intersecting all suspicious Sybil lists.

Almas Shehni *et al.* [24] proposed an algorithm to detect Sybil nodes in mobile WSNs which employs watchdog nodes and Hello packets exchanges between nodes. Each watchdog node uses a co-presence state diagram to provide partial information. A designated watchdog node aggregates all partial information and uses a detection rule to detect the Sybil nodes.

Jamshidi *et al.* [25] proposed an algorithm to detect Sybil nodes in mobile WSNs which uses watchdog nodes and a distributed labelling mechanism based on nodes' mobility. This algorithm consists of a monitoring phase and a detection phase, which are both performed by watchdog nodes. In the monitoring phase, watchdog nodes keep track of nodes' movement during $R$ rounds of network activities, cooperatively, and assign them bitwise labels. In the detection phase, each watchdog node discovers Sybil nodes based on the bitwise labels which assigned to sensor nodes. Since the movement patterns of all Sybil nodes belong to a malicious node is identical, their bitwise labels

| Algorithm | Network Type | Mechanism | Cooperation between Observer Nodes | Disadvantages |
|---|---|---|---|---|
| **[15]** | Static | Uses neighboring information and broadcasting two-hop packets | No Observers | High communication and memory overheads |
| **[16]** | Static | Uses neighbouring information and discovering common neighbours | No Observers | High communication overhead and low TDR |
| **[23]** | Mobile | Uses observer nodes to records the number of sensor nodes facing observer nodes and create separate sets | Exists | Low TDR, high FDR, high processing time for constructing and intersecting separate sets |
| **[24]** | Mobile | Employs a co-presence state diagram and watchdog nodes which overhear Hello packets exchanges between nodes | Exists | High communication and memory overheads |
| **[25]** | Mobile | Uses a distributed labelling mechanism to assign bit labels to nodes based on their movement | Exists | Low TDR, increasing memory overhead by growing the monitoring rounds ($R$) |
| **Proposed Algorithm** | Static | Uses mobile observer nodes and neighbouring information | Does Not Exist | Increasing memory overhead by growing the monitoring rounds ($R$) |

TABLE 1
Comparative analysis of various Sybil attack detection algorithms

are the same. This idea is used in the detection phase to mark Sybil nodes. This algorithm requires exchanging so many messages between the watchdog nodes which increases communication overheads and energy consumption as a result. Added to this, the algorithm has a relatively low Sybil nodes detection rate. Jamshidi *et al.* [26] also proposed a simple and precise algorithm to detect Sybil nodes in mobile WSNs which avoids observer nodes' cooperation to overcome the drawbacks of their previous algorithm [23], including low TDR, high FDR, and high processing overhead to the observer nodes.

Here, we present a comparative analysis of various Sybil attack detection algorithms (See Table 1). Algorithms [15] and [16] do not use observer nodes and they run on all nodes in the network. Hence, these algorithms impose high communication and memory overheads. On the other hand, algorithms like [23-25] utilize observer nodes, but they are not able to detect Sybil nodes in static WSNs because they rely on nodes' mobility. In addition, they suffer from some other drawbacks including high FDR (algorithm [23]), high memory/communication overhead (algorithm [24]), and low TDR (algorithm [25]). The proposed algorithm in this paper, only runs on some

observer nodes and therefore, does not impose overheads on all nodes in the network. Furthermore, the observer nodes in the proposed algorithm, unlike algorithms [23–25], work independently which reduces communication/processing overheads.

## 3 ASSUMPTIONS

### 3.1 System Assumptions

Sensor network contains two sets of SN (sensor nodes) and MN (mobile observer node). SN nodes are stationary and their task is to perform network mission. MN nodes are mobile throughout the network's life and their task is to identify Sybil nodes. Nodes are distributed randomly in a two-dimensional area and are not aware of their location. Mobile observer nodes move in the network during the network's life according to mobility models like random waypoint. It is assumed that observer nodes move to a new location (which is chosen randomly) after each time period $t$. each node has a unique identity. Radio range of all nodes is constant and equal to $r$. it is assumed that mobile observer nodes are aware of network's approximate density, $d$ (or the average number of neighbors of a node), and if network's density changes, the base station informs mobile observer nodes securely. It is assumed that the nodes communicate with each other via a wireless radio channel and broadcast information in an Omni-directional mode. It is also assumed that the sensor network is deployed in adversary environment, thus, the network is not secure and nodes might be captured by the adversary. Normal sensor nodes are not tamper-resistant and if the adversary captures a node, it can access its secret information and reprogram it. But it is assumed that mobile observer nodes are robust against interference and if they are captured by an adversary, they cannot be decoded and reprogrammed.

### 3.2 Attack Model

In this paper, according to taxonomies of [6], direct, simultaneous Sybil attack and fake or stolen identities are considered. The node captured by the adversary is called malicious node and other nodes of the network are called normal nodes. Each malicious node propagates $S$ identities (Sybil nodes).

According to [15], it is also assumed that Sybil nodes propagated by a malicious node are more than normal neighbor nodes in the neighborhood of a node ($S > d$). This assumption is made by considering the point that the adversary can only disrupt network operation by establishing Sybil attacks if it injects many identities into the network. Therefore, if $S$ is a small value, the adversary has to capture many normal nodes and reprogram them as malicious nodes. But as mentioned in [15], doing so would be difficult and time-consuming for the adversary, because it has to capture, decode, reprogram and control a great number of nodes. Therefore, adversary prefers to

capture less normal nodes, but each captured node propagates a lot of fake identities.

## 4 PROPOSED ALGORITHM

As mentioned before, a malicious node which establishes a Sybil attack, propagates $S$ fake identities simultaneously (which are called Sybil nodes). In addition, assuming $S > d$, it can be concluded that existence of a Sybil attack in a specific location, $L$, of the network, causes the number of neighbors around area $L$ to be more than average, $d$. The main idea of the proposed algorithm is also adopted from this issue. That is, if the number of nodes in an area of the network is more than $d$, Sybil attack might have been established in that area. Thus, these areas (which are called suspicious area hereafter) should be searched for Sybil nodes. Flowchart of the proposed algorithm is shown in Figure 1.

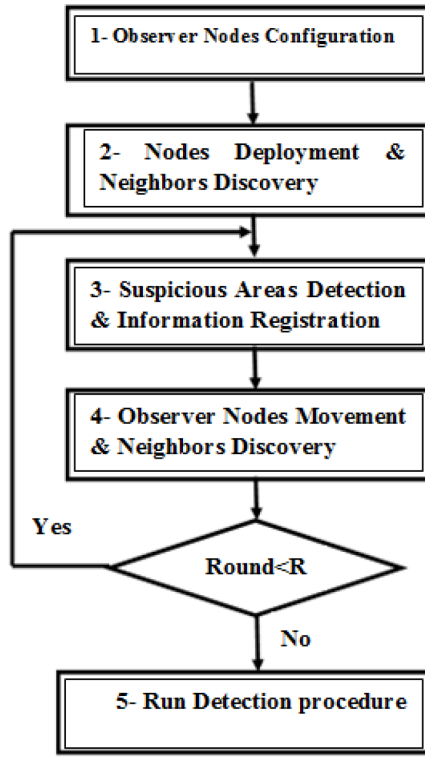


FIGURE 1
Flowchart of the proposed algorithm.

In step 1, before deploying nodes in the network, observer nodes are configured. That is, the proposed algorithm is loaded on these nodes (observer nodes). In addition, parameters $R$ and $d$ of these nodes are also tuned. As mentioned before, $d$ is the approximate density of the network and $R$ is the number of observer nodes' walking rounds for identifying Sybil in the network. Hereafter, each walking step of the observer nodes is called round. In each round, observer nodes determine a destination and move toward that destination. Then, they stop for $t$ time units in the destination and monitor the surroundings of the destination. Then they go to the next round.

In step 2, all nodes are deployed randomly. After deployment in the network, observer nodes remain in their location for $t$ time units. Immediately after deployment of nodes in the environment, each node broadcasts a "Hello" message in order to discover neighboring nodes. Each node stores its neighbors' list in a vector called *neighborList*. It should be noted that each malicious node broadcasts a "Hello" message for each Sybil identity to reveal all of its Sybil nodes in the network. In this step, observer nodes discover their current neighbors easily (according to "Hello" messages broadcasted by nodes).

In step 3, observer nodes try to detect suspicious areas and record information of these areas in their memory. Each observer node has an MSR matrix in its memory in which it records information regarding probable suspicious areas. After discovering current neighbors, observer nodes investigate if they are located in a suspicious area or not. If the number of observer node's neighbors is more than $2d$ (or $d + d/2$), it considers the current area as a suspicious area, and thus it records its current neighbors' list, *neighborList*, in a row of MSR matrix. This is the end of round 1.

In step 4, next round (round 2) begins. That is, observer nodes determine a destination randomly and move towards that destination. In other words, observer nodes will go through one walking round in the network. After reaching the destination, observer nodes remain there for $t$ time units. During this time period, observer nodes try to discover their current neighbors again. But discovery procedure in this step is different and more difficult compared to round 1, because in round 1, after deployment in the environment, nodes broadcast a "Hello" message. Therefore, observer nodes can easily discover their neighbors. But in next rounds, observer nodes have to employ another approach for discovering their neighbors. Since in subsequent rounds, nodes do not broadcast a "Hello" message (Hello message is broadcasted only once, because nodes are stationary). After round 1, observer nodes can discover their neighbors using two approaches:

In the first approach, whenever an observer node moves to a new location, it broadcasts a request and waits for the response from neighbors. Observer node records responder's identity as a current neighbor in its *neighborList*.

The problem of this approach is that if there are any Sybil nodes in that area, they might not respond to the observer node, thus suspicious nodes cannot be detected.

In the second approach, the observer node, requests its neighbors, e.g. $u$, to send a packet containing the identity of the node ($u$) and its neighbors' list (neighbors of $u$). Since each node, discover its neighbors (normal and Sybil) after deployment in the environment, it can send this *neighborList* to observer nodes, if necessary. Observer node selects a list of responder nodes and only some nodes of the *neighborList*s received from its neighbors as the list of neighbors. There is certainly a list of nodes in *neighborList*s which are not one-hop neighbors for observer node, but they are two-hop neighbors and they should not be considered as the observer node's neighbors. Here, we propose a simple probabilistic approach for the observer node for either selecting or not selecting a node $v$ in a responder node's *neighborList*. If the identity of $v$ is repeated in half of the *neighborList*s, observer node considers and selects $v$ as its neighbor. Using this method, even if Sybil nodes do not respond the observer, the observer node becomes aware of Sybil nodes, because the identity of these Sybil nodes is stored in the *neighborList* of normal nodes located in this area of the network.

After that, the observer nodes discovered their neighbors in the current step, if the number of rounds is smaller than or equal to $R$, discovering probable suspicious areas and recording information (stage 4) runs again and then the next round begins. In other words, steps 3 and 4 of the proposed algorithm is repeated $R$ times. Thus, after $R$ rounds, each observer node will have $0 \leq P \leq R$ individual lists of nodes' identities, where each list belongs to a suspicious area. It should be noted that a malicious node might create several suspicious areas. As it can be seen in Figure 2, there is a malicious

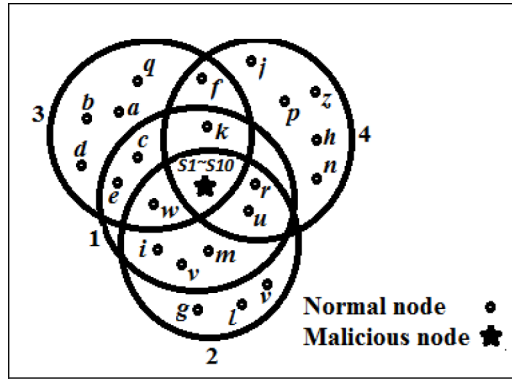


FIGURE 2
Suspicious areas around a malicious node.

node which broadcasts 10 fake identities (S1 to S10) and each circle (or area) contains 9 legal nodes and 10 Sybil nodes. Assuming $d = 10$, if observer node is at the center of each circle since the number of its neighbors is more than $d + d/2$, this area is considered as suspicious.

After going through $R$ rounds in the network environment and recording information in MSR, observer nodes perform detection procedure (step 5). In this step, each observer node detects Sybil nodes individually using MSR information. One option for the observer node is to mark all nodes in MSR as Sybil nodes. But this detects many normal nodes in the neighboring of a malicious node as Sybil nodes incorrectly. Thus, the false detection rate of the proposed algorithm increases. One better option is that the observer node filters its MSR matrix. Here, a filtering algorithm is proposed and it pseudo-code is presented in Figure 3.

The proposed filtering algorithm accepts an MSR matrix as input and return a list of Sybil identities as output. It's supposed MSR has $P$ rows. Each

```
Function filtering (MSR)
Input: a matrix contains P rows and each row contains a vectors of node identities
Output: a list of Sybil nodes
i←1
while i ≤ number of rows in MSR
    Cnt←0
    for each vector j>i in MSR
        if the number of common items in vectors i and j is equal to or greater than the threshold T_s then
            Cnt←Cnt+1
        end if
    end for
    if Cnt < threshold T_c then
        remove vector i from MSR
    else
        for each vector j>i in MSR
            temp← the common items in vectors i and j
            if the number of items in temp is equal to or greater than the threshold T_s then
                vector i← temp
                remove vector j from MSR
            end if
        end for
    end if
end while
return all items in MSR as Sybil nodes
```

FIGURE 3
Pseudo-code of the proposed filtering algorithm.

| Suspicious area number | Suspicious identities |
|---|---|
| 1 | $S1, S2, ...S10, c, e, i, k, m, r, u, v, w$ |
| 2 | $S1, S2, ...S10, g, i, l, m, r, u, v, w$ |
| 3 | $S1, S2, ...S10, a, b, c, d, e, f, k, q, w$ |
| 4 | $S1, S2, ...S10, f, j, h, k, n, p, r, u, z$ |
| ... | .. |
| $P$ | ... |

TABLE 2
The typical MSR matrix for an observer node

row $i$ of MSR contains a vector of node identities, called vector $i$. The filtering process will be started for row $i = 1$. For each vector $i$ in the MSR,the number of other vectors ($j > i$) which have at least $T_s$ common items with vector $i$ is first counted. This quantity is stored in a variable, named *Cnt*. Then, vector $i$ will be removed from MSR if *Cnt* is less than the threshold $T_c$ (this condition guarantees that the observer node marks a node as a Sybil node if it has appeared at least $T_c + 1$ times in areas around a malicious node. Thus, less normal nodes are marked as Sybil nodes incorrectly). Otherwise, the filtering process of vector $i$ will be continued. In this case, vector $i$ is checked with all vectors $j > i$, sequentially. First, the common items of vectors $i$ and $j$ are extracted and stored in a temporary vector, named *temp*. Then, vector *temp* stored in vector $i$ and vector $j$ is removed from MSR, if the number of items in vector *temp* is equal to or greater than $T_s$. Since vector $i$ and vector $j$ probably contain a list of nodes which are available in suspicious areas around a malicious node, it is better just to preserve intersection of these lists to mark less normal nodes as Sybil nodes. This procedure will be repeated for all vectors in MSR. When the filtering algorithm is finished, observer node marks all nodes which are remained in MSR matrix as Sybil nodes.

For example, consider Figure 2 and assume $d = 10$, $T_c = 3$, $T_s = 10$. In addition, assume that observer node has traveled around malicious node exactly in four regions specified in Figure 2 throughout $R$ rounds. Thus, MSR matrix of the observer node contains information which is shown in Table 2.

Now, observer node compares the first row of its MSR matrix with other rows while filtering; since the first row has at least $T_s = 10$ common items with rows 2, 3 and 4, the intersection of these rows is kept in row 1 and rows 2,3 and 4 are eliminated. Thus, only nodes S1 to S10 remain in row 1 and are marked as Sybil nodes. Now, if the observer node has only appeared in 3 areas around this malicious node (like areas 1, 2 and 3), it would not mark any nodes as Sybil node, because *Cnt* would not be greater than or equal to $T_c$ for any row. Therefore, all rows of MSR matrix would be eliminated. It can

be seen that in such situation, Sybil nodes are not detected by observer node. But if $T_c = 2$ and the observer node appears only in areas 1, 2 and 3 around the malicious node, the intersection of these areas would be marked as Sybil nodes. This shows that by increasing $T_c$, not only the detection rate of Sybil nodes decreases but also less normal nodes would be marked as Sybil nodes incorrectly. The result of experiment 2 in the next section shows this issue.

## 5 EFFICIENCY EVALUATION AND SIMULATION RESULTS

In this section, memory, communications, and computations overhead of the proposed algorithm are evaluated first and then the efficiency of the proposed algorithm is compared with other existing algorithms through some experiments.

### 5.1 Evaluating Overhead of the Proposed Algorithm

**Memory Overhead:** since the proposed algorithm is executed by the observer nodes, thus memory overhead is only related to observer nodes and no extra overhead is imposed on normal nodes. In the proposed algorithm, each observer node requires some memory space for storing its MSR matrix. As mentioned before, if observer nodes enter a suspicious area while walking, that is, the number of nodes in this area is more than $d$, list of nodes located in this area is stored in MSR matrix. Assuming that observer nodes walk in the network environment $R$ rounds, a $R \times E \times \alpha \times d$ space of the memory would be required for storing MSR matrix. Here, $E$ is the probability that the observer node is located in one of the suspicious areas in any walking round. $E$ is calculated in Appendix A. $\alpha \times d$ is the number of nodes in a suspicious area. $2 < \alpha \leq M$ and $M$ is the number of total malicious nodes in the network. Maximal state, where $\alpha = M$, occurs when all $M$ malicious nodes are located simultaneously in the neighborhood of the observer node. As $M$ increases, the probability of such state decreases. In addition, each observer node requires $\alpha \times d$ space for storing *neighborList* in each walking round. Therefore, the memory overhead of the proposed algorithm for normal sensors is zero and for observer nodes, memory overhead is of order $O(R \times E \times \alpha \times d)$.

**Communications Overhead:** communications overhead of the proposed algorithm is related to walking rounds of observer nodes. Each observer node, in each walking round, issues a request message. Nodes located in this area respond to the observer node's request (a packet including identity and *neighborList*). Therefore, in one round, $|MN| + |MN| \times d$ packets are transmitted in the whole network. $|MN|$ is the number of observer nodes. Thus, total

communication overhead imposed on the network after $R$ walking rounds would be of order $O(R \times |MN| \times d)$.

**Computation Overhead:** in the proposed algorithm, the computation overhead is only imposed on observer nodes. In fact, computation overhead of the proposed algorithm is associated with detection of Sybil nodes after $R$ walking rounds are finished (filtering algorithm of MSR matrix). Assuming that MSR matrix of each observer node contains $P$ rows ($P \leq R$), each row $i$ of the matrix should be compared with $j > i$ rows and if there are any intersections among lists of these rows which is greater than or equal to $T_s$, the common items of these rows is replaced in row $i$. Since the number of items in each row of MSR matrix is approximately $2d$, the intersection of rows $i$ and $j$ would be of order $O(d \log d)$. First, each list is ordered with $O(d \log d)$ time and then the intersection of these lists is ordered with $O(d)$ time. Therefore, computation overhead imposed on each normal sensor node is zero and each observer node would be of order $O(P^2 \times d \log d)$

### 5.2 Simulation Results

The proposed algorithm is implemented with JSIM simulator [27] and several experiments are performed in order to evaluate its efficiency; obtained results are compared with some other existing algorithms. In a general classification, algorithms for identifying the Sybil attack can be categorized into two classes:

1. Algorithms which are executed in one round. In such classes, the algorithm is executed just once and in a specified time, such as the proposed algorithm and algorithms [15] and [16].
2. Algorithms which are executed in several rounds (Here, $R$ rounds). Such algorithms are executed periodically in regular time intervals (during network's life), such as algorithms [23–25].

Here, the proposed algorithm is compared with both classes existing algorithms. For this purpose, in experiments where the number of execution rounds is an evaluation parameter, like experiment 1, the proposed algorithm is compared with algorithms of class 2 and in experiments where the number of execution rounds is not an evaluation parameter, like experiment 3, the proposed algorithm is compared with algorithms of both classes. Thus, in each experiment, different algorithms are compared. Our evaluation metrics are as follows:

- **True Detection Rate (TDR):** percentage of Sybil nodes which are detected by a security algorithm.

- **False Detection Rate (FDR):** percentage of normal nodes which are detected as Sybil nodes incorrectly.
- **Average True / False Detection Rate:** in order to calculate this measure, the proposed algorithm is executed for all states (variations of different parameters like total number of nodes, number of malicious nodes and . . .) and possible conditions and the obtained results are averaged.

In simulations, it is assumed that the network includes $N$ sensor nodes where $M$ out of $N$ nodes are malicious nodes and $q = |MN|$ is the number of mobile observer nodes. Nodes are distributed in a $100 * 100m^2$ area randomly. Each malicious node broadcasts $S$ fake identities. Radio range of all nodes (normal, malicious and observer) is the same and equal to 10 meters. In all experiments, $T_s$ is tuned with $d$.

In order to validate experiments, each simulation is repeated 50 times and the final result is obtained by averaging these 50 times.

**Experiment 1:** in this experiment, the efficiency of the proposed algorithm is evaluated in terms of TDR and FDR and the obtained results are compared with results obtained from other algorithms. In this experiment, parameters are considered to be as $M = 5, q = 4, S = 20, T_C = 2, N = 300$ and FDR and TDR of the proposed algorithm are compared with algorithm [23] (with parameter $T_S = 6$) and algorithm [25] (with parameter $T_{min} = 10$). $R$ is changed within the range [20 to 180]. Figure 4 shows the TDR and Figure 5 shows the FDR of the experiment.

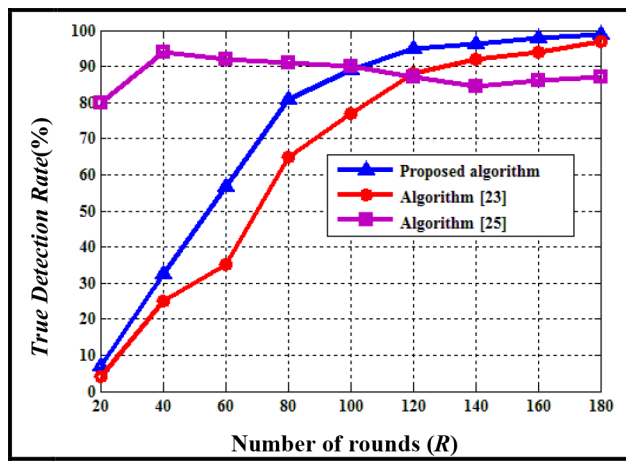


FIGURE 4
Comparing TDR of the proposed algorithm with other algorithms for different values of $R$.

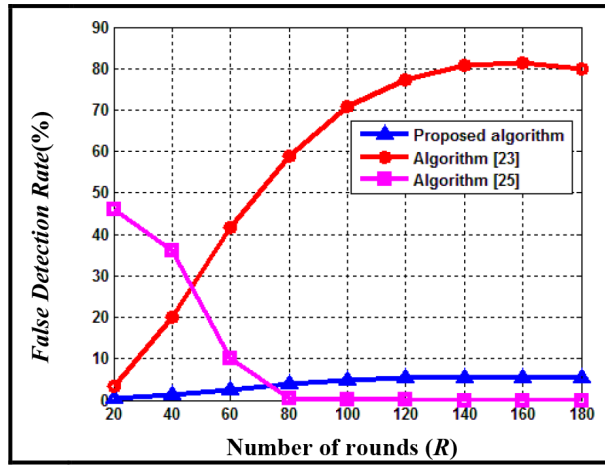


FIGURE 5
Comparing FDR of the proposed algorithm with other algorithms for different values of $R$.

The results of this experiment show that the TDR of the proposed algorithm, for $R = 100$ and $R = 180$ is 90% and 99% respectively. In addition, the TDR of the proposed algorithm for different values of $R$ is greater than that of algorithm [25] and for $R > 100$, TDR is greater than that of algorithm [23].

Furthermore, Figure 5 shows that for $R \leq 100$, the FDR of the proposed algorithm is less than 5% and for $R > 100$, FDR is about 5%. Figure 5 also shows that for $R < 180$, FDR of the proposed algorithm is less than the other algorithms. Indeed, the performance of the algorithm [23] in decreasing FDR is poor in such a way that for $R > 1000$, FDR is less than 10%. In addition, results show that by increasing $R$, FDR of the proposed algorithm increases a little, such that increasing $R$ from 20 to 120, increases FDR from 0% to 5%. But for $R > 120$, FDR remains stable on 5%. The reason is that in the proposed algorithm, sensor nodes which are located close to a malicious node, and are known as unlucky legal nodes, might be detected as Sybil nodes incorrectly. But depending on the density of the network and number of malicious nodes in the network, the number of these unlucky legal nodes is limited, thus maximum FDR is also limited. Experiment result in Figure 5 also shows this issue. It can be seen in this experiment that after 120 rounds, approximately all unlucky legal nodes (5% ) are detected as Sybil nodes incorrectly and for $R > 120$, approximately no legal node is detected as Sybil node incorrectly, thus FDR remains constant.

Considering the results of this experiment, an optimum point, $R_{opt}$, can be considered such that both FDR and TDR of an algorithm are more suitable

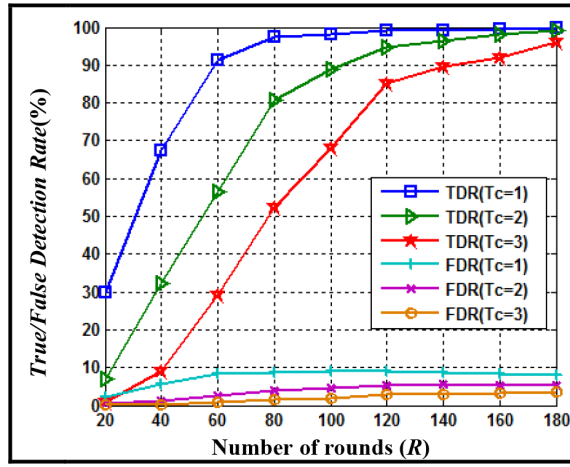


FIGURE 6
Effect of parameter $Tc$ on TDR and FDR of the proposed algorithm.

compared to other points. For algorithm [23], in $0 < R < 180$, best tolerable point is $R_{opt} = 40$ which results in TDR $= 25\%$ and FDR $= 20\%$. As mentioned before, algorithm [23] performs very poor in decreasing FDR and optimum point of this algorithm can be $R_{opt} = 1000$ which results in TDR $=$ 100% and FDR $= 9\%$. For algorithm [25], $R_{opt} = 80$ which results in TDR $= 91\%$ and FDR $= 0\%$. For the proposed algorithm, the optimal point is $R_{opt} = 180$ which results in TDR $= 99\%$ and FDR $= 5\%$.

**Experiment 2:** this experiment investigates the effect of parameter $T_c$ on detection accuracy of the proposed algorithm. In this experiment, parameters $M = 5, q = 4, S = 20, N = 300$ are considered and TDR and FDR of the proposed algorithm for $T_c = 1, 2, 3$ and $R = 120, 180$ are evaluated and the obtained results are shown in Figure 6. As mentioned before, in Sybil nodes detection phase (after $R$ rounds), each observer node should filter its MSR matrix and eliminate row $i$, if there is less than $T_c$ rows such that there are at least $T_s$ common items with row $i$. It is clear that by increasing $T_c$, both TDR and FDR will decrease. $T_c = k$ means that each observer node should have appeared at least $k + 1$ times in suspicious areas around a malicious node and have recorded a list of nodes located in these areas in its MSR matrix in order not to eliminate them in filtering step. Therefore, the greater is $k$, the probability that the observer node appears at least $k + 1$ times in suspicious areas around a malicious node during $R$ rounds is less, and therefore TDR and FDR also decrease. On the contrary, the smaller is $k$, TDR and FDR would increase. Experiment results in Figure 6 show this issue.

| | $S = 8$ | | $S = 12$ | | $S = 16$ | | $S = 20$ | |
|---|---|---|---|---|---|---|---|---|
| **Algorithm** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** |
| Algorithm [15] | 98.8 | 18 | 99.5 | 9 | 99.7 | 6 | 99.9 | 3.7 |
| Algorithm [23] | 80.9 | 72 | 80.6 | 69 | 80.8 | 64 | 78.6 | 61 |
| Algorithm [25] | 85.5 | 0.18 | 92.8 | 0.13 | 93.1 | 0.04 | 94.3 | 0.03 |
| Proposed Algorithm | 62.6 | 6.3 | 71 | 4.7 | 89.6 | 4.8 | 90 | 4.5 |

TABLE 3
Comparing TDR and FDR of the proposed algorithm with other algorithms for different values of $S$ and $R = 100$.

| | $S = 8$ | | $S = 12$ | | $S = 16$ | | $S = 20$ | |
|---|---|---|---|---|---|---|---|---|
| **Algorithm** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** |
| Algorithm [15] | 98.8 | 18 | 99.5 | 9 | 99.7 | 6 | 99.9 | 37 |
| Algorithm [23] | 96.7 | 84.9 | 96 | 76 | 95.9 | 67 | 96.7 | 59 |
| Algorithm [25] | 83.5 | 0 | 85 | 0 | 89.8 | 0 | 90.4 | 0 |
| Proposed Algorithm | 84.4 | 12.5 | 86.8 | 6.2 | 98.2 | 5.6 | 97.4 | 5.3 |

TABLE 4
Comparing TDR and FDR of the proposed algorithm with other algorithms for different values of $S$ and $R = 160$.

**Experiment 3:** this experiment investigates the effect of parameter $S$ on detection accuracy of the proposed algorithm and the obtained results are compared with the results obtained from other algorithms. In this experiment, parameters $M = 5, q = 4, T_c = 2$, and $N = 300$ are considered and efficiency of the algorithm is evaluated for $S = 8, 12, 16, 20$. Table 3 shows the results of this experiment for $R = 100$ and Table 4 shows results for $R = 160$. Algorithm [15] is set with $\theta = 0.7N$ and other parameters of the algorithms are set as in experiment 1. As mentioned before, the main idea of this algorithm is based on this assumption that the number of Sybil nodes broadcasted by a malicious node is more than the number of normal neighbors among neighbors of a node. Therefore, the smaller is $S$ compared to $d$, TDR decreases, and FDR increases because observer nodes cannot detect suspicious areas while walking.

**Experiment 4:** the purpose of this experiment is to evaluate the effect of $q$, the number of observer nodes, on the efficiency of the proposed algorithm and compare the obtained results with other algorithms. Algorithms [23] and [25] also rely on observer nodes. In this experiment, parameters $M = 5, T_c = 2, S = 14$, and $N = 300$ are considered; TDR and FDR of the proposed algorithm and other algorithms are evaluated for $q = 4, 6, 7$ and values of $R$ between 20 to 100. Figure 7 and Figure 8 show results of

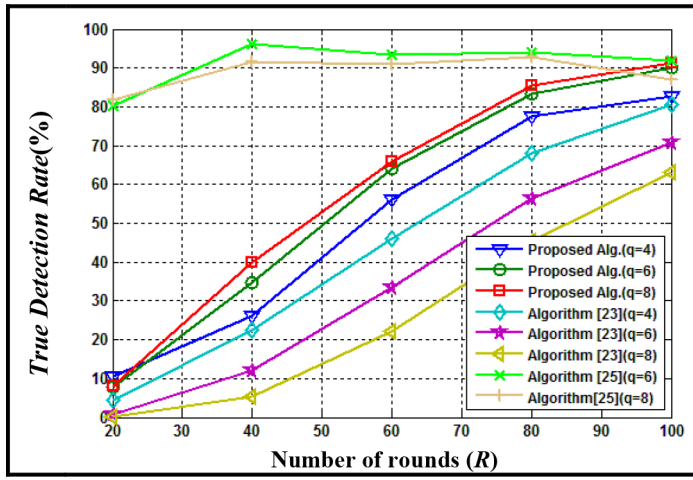


FIGURE 7
Effect of parameter $q$ (number of observer nodes) on TDR of the proposed algorithm and other algorithms.

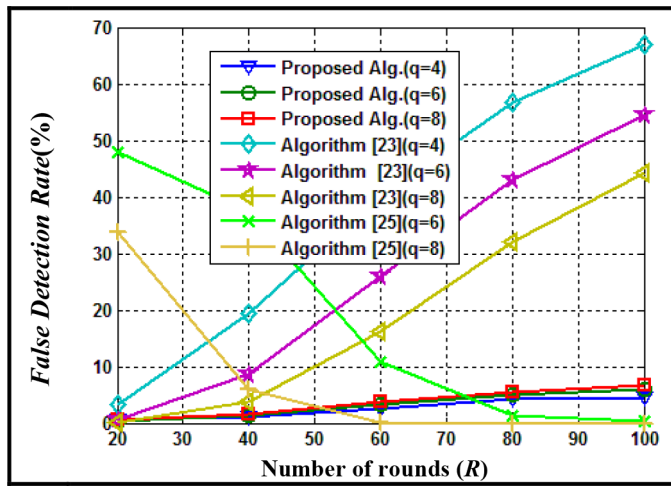


FIGURE 8
Effect of parameter $q$ (number of observer nodes) on FDR of the proposed algorithm and other algorithms.

TDR and FDR, respectively. It can be seen from the results that the TDR of the algorithm [23] decreases by increasing the number of observer nodes because, in this algorithm, observer nodes cooperate with each other to detect Sybil nodes. The same holds for algorithm [25], that is, increasing number

of observer nodes decreases TDR. In algorithm [25], observer nodes cooperate to detect Sybil nodes. Indeed, as shown in Figure 8, decreasing number of observer nodes in algorithms [23] and [25] will increase FDR. But in the proposed algorithm, observer nodes detect Sybil nodes independently. Thus, increasing number of observer nodes will increase both TDR and FDR. As it can be seen in Figure 8, FDR of the proposed algorithm grows negligibly by increasing number of observer nodes, which demonstrates the superiority of the proposed algorithm compared to other two algorithms. In addition, the result of this experiment in Figure 7 shows that TDR of the proposed algorithm for different values of $q$ is always better than algorithm [23] and for $R > 100$ it is better than algorithm [25]. In terms of FDR, the proposed algorithm is superior to algorithm [23] and for $R < 50$, it is better than algorithm [25], but by increasing $R$, the FDR of algorithm [25] will tend towards 0% while the FDR of the proposed algorithm increases and becomes 5%.

**Experiment 5:** this experiment investigates the effect of the number of malicious nodes, $M$, on detection accuracy of the proposed algorithm and obtained results are compared with algorithms [23] and [25]. In this experiment, parameters $q = 4$, $T_c = 2$, $S = 14$, and $N = 300$ are considered and efficiency of the proposed algorithm for $M = 1, 5, 10$ for values of $R$ between 20 to 160 is evaluated and the results are shown in Figure 9 and

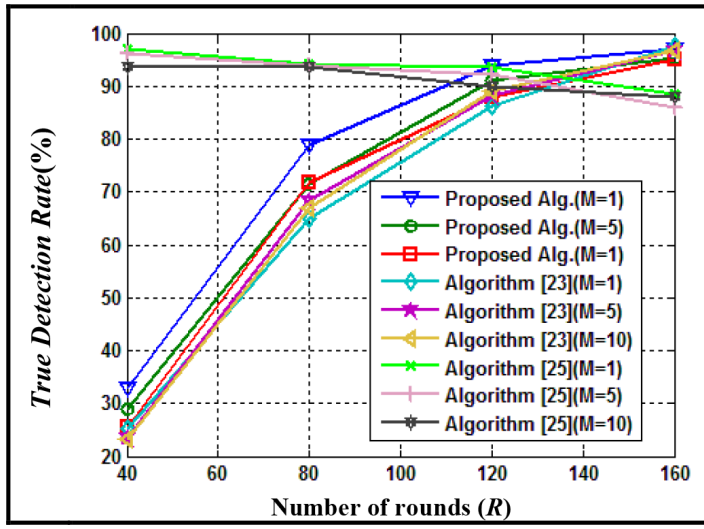


FIGURE 9
Effect of parameter $M$ (number of malicious nodes) on the TDR of the proposed algorithm and other algorithms.

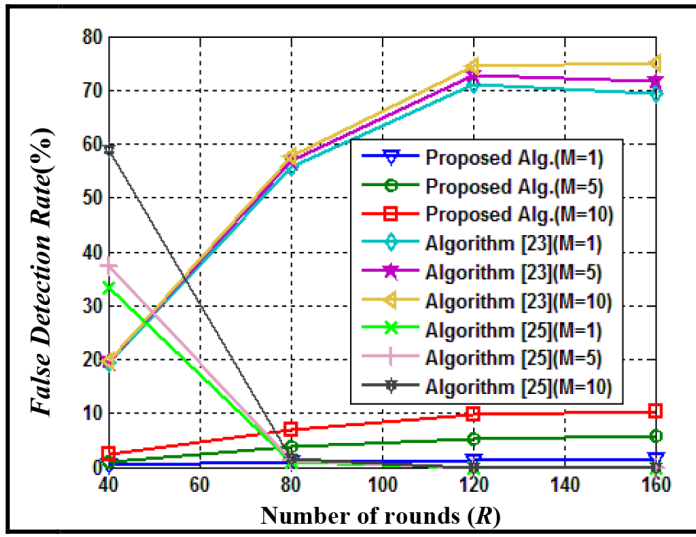


FIGURE 10
Effect of parameter $M$ (number of malicious nodes) on the FDR of the proposed algorithm and other algorithms.

Figure 10. As can be seen from the results shown in Figure 10, by increasing the number of malicious nodes in the network, the TDR of the proposed algorithm and the other algorithms also decrease. This is because, by increasing the number of malicious nodes in the network, more suspicious nodes are created in the network. Thus the probability that observer nodes meet suspicious areas around each malicious node at least $T_c + 1$ times becomes smaller. Therefore, TDR decreases. On the other hand, increasing number of malicious nodes increases FDR of the proposed algorithm and algorithms [23] and [25] (Figure 10). This is because if a normal node is located in the close neighborhood of a malicious node, it is probably detected as a Sybil node. Therefore, by increasing the number of malicious nodes in the network, the probability that normal nodes are located in the close neighborhood of a malicious node, increases. Thus, FDR increases.

**Experiment 6:** the purpose of this experiment is to investigate the effect of the number of nodes, $N$, on the efficiency of the proposed algorithm. In this experiment parameters $M = 5, T_c = 2, S = 14, q = 4$, and $N = 100, 200, 300, 400$ are set and FDR and TDR of the proposed algorithm are evaluated. Table 5 shows the results of this experiment for $R = 100$ and Table 6 shows results for $R = 160$. It can be seen that by increasing the number of nodes, TDR of the proposed algorithm decreases. That is because

| | $N = 100$ | | $N = 200$ | | $N = 300$ | | $N = 400$ | |
|---|---|---|---|---|---|---|---|---|
| **Algorithm** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** |
| Algorithm [23] | 81.3 | 40 | 80.4 | 60 | 80.6 | 66.9 | 81 | 70 |
| Algorithm [25] | 88 | 0 | 90 | 0.1 | 91.8 | 0.3 | 91.1 | 0.1 |
| Proposed algorithm | 90.4 | 13.2 | 86.6 | 4.6 | 83.8 | 5 | 51.8 | 3.2 |

TABLE 5
Comparing TDR and FDR of the proposed algorithm with other existing algorithms for different values of $N$ and $R = 100$.

| | $N = 100$ | | $N = 200$ | | $N = 300$ | | $N = 400$ | |
|---|---|---|---|---|---|---|---|---|
| **Algorithm** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** | **TDR** | **FDR** |
| Algorithm [23] | 96.9 | 24 | 96.4 | 58 | 97 | 71 | 96.5 | 80 |
| Algorithm [25] | 88.4 | 0 | 88.1 | 0 | 86.6 | 0 | 89.8 | 0 |
| Proposed algorithm | 99.2 | 15.9 | 99.4 | 6.3 | 95 | 5.6 | 68.2 | 5 |

TABLE 6
Comparing TDR and FDR of the proposed algorithm with other existing algorithms for different values of $N$ and $R = 160$.

of the increasing number of nodes in the network, density (average number of neighbors of a node), $d$, increases. This contradicts $S > d$, thus TDR decreases. For example, Table 6 shows that TDR of the proposed algorithm for $N = 100$ is 99.2%, while this decreases to 68.2% for $N = 400$.

It is clear that if $S$ is increased (for example to $S = 20$), the TDR of the proposed algorithm will be higher than 68.2%. To prove this, this experiment is evaluated once again for parameters $q = 4$, $T_c = 2$, $S = 20$, $M = 20$ and $N = 150, 250, 350, 450$. Figure 11 shows the results of this experiment. Since in this experiment, $S > d$ for all $N$, thus TDR of the proposed algorithm would be high. The result of this experiment showed that for $N < 350$, TDR is higher than 98% and for $N = 450$, TDR is 94%. In addition, the results showed that FDR of the proposed algorithm for different values of $N$ is 22%. It should be noted that in this experiment, the number of malicious nodes is $M = 20$. It was shown in experiment 5 that by increasing the number of malicious nodes in the network, the FDR of the algorithm would also increase.

**Experiment 7:** in this experiment, the efficiency of the proposed algorithm and other existing algorithms is evaluated in terms of average TDR, average FDR, memory overhead, and communication overhead. In this experiment we set $R = 180$ for algorithms [23–25] and the proposed algorithm. The

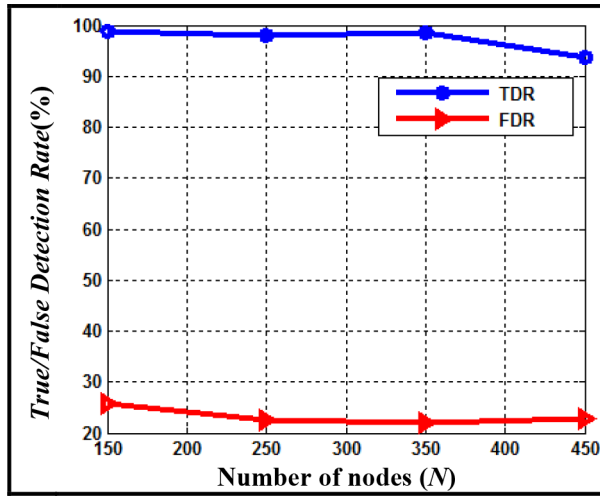


FIGURE 11
Effect of parameter $N$ (number of nodes) on TDR and FDR of the proposed algorithm for $R = 160$ and $S = 20$.

evaluated results are shown in Table 7. The results show that average TDR of the proposed algorithm and algorithm [24] is 99%, algorithm [16] is 98%, algorithm [23] is 96%, and algorithm [25] is 94%. Also, the results show that the average FDR of the proposed algorithm and algorithm [15] is 5%, algorithm [16] is 2.5%, algorithm [23] is 30%, and algorithms [24] and [25] is 0%.

## 6 CONCLUSION

In this paper, a novel algorithm employing mobile observer nodes was proposed for identifying Sybil nodes in stationary wireless sensor networks. In this algorithm, observer nodes first walk in the network to identify suspicious areas and record list of nodes located in such areas in a matrix called MSR in their memory. Then each node filters its MSR matrix individually and detects Sybil nodes. The proposed algorithm is simulated and its efficiency is compared with some other existing algorithms. Experiment results demonstrate the desired performance of the proposed algorithm in terms of true detection rate and false detection rate. As mentioned before, in the proposed algorithm, observer nodes detect Sybil nodes individually and this decreases the detection rate in some conditions. In future works, the detection rate of Sybil nodes in the last phase of the algorithm can be increased by employing a cooperation mechanism among observer nodes.

| Algorithm | Average TDR | Average FDR | Memory overhead (per network) | Communication overhead (per network) |
|---|---|---|---|---|
| [15] | 99% | 5% | $N \times d^2$ | $N \times d^2$ |
| [16] | 98% | 2.5% | $N \times d^2$ | $N \times (2d+1)$ |
| [23] | 96% | 30% | $\|MN\| \times N$ | $\|MN\| \times (\|MN\|-1) \times k$ |
| [24] | 99% | 0% | $\|MN\| \times (N + M \times S)^2$ | $(N + M \times S)^2$ |
| [25] | 94% | 0% | $N \times R \times \left\lceil \log_2^{\|MN\|} \right\rceil$ | $R \times 2 \times \|MN\|$ |
| Proposed Algorithm | 99% | 5% | $R \times E \times \alpha \times d$ | $R \times \|MN\| \times d$ |

TABLE 7
Comparative analysis of various Sybil attack detection algorithms

$N$ : the total number of nodes in the network
$d$ : the average number of nodes' neighbors
$|MN|$ : the total number of observer/watchdog nodes in the network
$M$: the number of malicious nodes in the network
$S$: the number of Sybil identities propagate by each malicious node
$R$: the number of rounds in which the algorithm should monitor the nodes' traffic and mobility
$E$: the probability that the observer node is located in a suspicious area in any of the walking rounds.
$k$ : is the network diameter
$\alpha$ : is a number between 2 and $M$

## APPENDIX A: CALCULATING E

This section calculates $E$, the probability that the observer node is located in a suspicious area in any of the walking rounds. If there is a malicious node in the network, whenever an observer node is located in the neighborhood of this malicious node, it has entered a suspicious area. In other words, if the observer node enters an area of $r^2\pi$ around the malicious node, it has entered a suspicious area. Therefore, assuming there is a malicious node in an operational environment of $x * y$ and the radio range of $r$ for nodes, the probability that this observer node is located in a suspicious area in any of the walking rounds would be:

$$E = \frac{r^2\pi}{X \times Y} \tag{A.1}$$

And if there are $M$ malicious nodes in the network, the maximum value of $E$ is obtained when there are no neighboring malicious nodes. Thus we have:

$$E = Min\left\{1, \quad \frac{M \times r^2\pi}{X \times Y}\right\} \tag{A.2}$$

## REFERENCES


[1] Ammari, H.M., Gomes, N., Jacques, M., AXIM, B.M., Yoon, D. (2015). A Survey of Sensor Network Applications and Architectural Components. *Adhoc & Sensor Wireless Networks.* 25(1-2), pp. 1–44.

[2] Jamshidi, M., Shaltooki, A. A., Zadeh, Z. D., Darwesh, A. M. (2018). A Dynamic ID Assignment Mechanism to Defend Against Node Replication Attack in Static Wireless Sensor Networks. *JOIV: International Journal on Informatics Visualization*. 3(1).

[3] Douceur, J.R. (2002). The Sybil attack. *International Workshop on Peer-to-Peer Systems*, Springer, Berlin, Heidelberg, pp. 251–260.

[4] Karlof, C., Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures. *Ad hoc networks*, 1(2), pp. 293–315.

[5] Gkountis, C., Taha, M., Lloret, J., Kambourakis, G. (2017). Lightweight algorithm for protecting SDN controller against DDoS attacks. In 10th IFIP Wireless and Mobile Networking Conference (WMNC),IEEE, Valencia, Spain, 25–27 September, pp. 1-6.

[6] Newsome, J., Shi, E., Song, D., Perrig, A. (2004). The Sybil attack in sensor networks: analysis & defenses. Proceedings of the 3rd international symposium on Information processing in sensor networks. Berkeley, California, USA, April 26–27, pp. 259–268. ACM.

[7] Liu, D., Ning, P., Li, R. (2005). Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security* (TISSEC), 8(1), pp. 41–77.

[8] Zhong, S., Li, L., Liu, Y.G., Yang, Y.R. (2004). Privacy-preserving location-based services for mobile users in wireless networks. Department of Computer Science, Yale University, Technical Report ALEU/DCS/TR-1297.

[9] Demirbas, M., Song, Y. (2006). An RSSI-based scheme for Sybil attack detection in wireless sensor networks. Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks, *IEEE Computer Society*, June 26–29, pp. 564–570.

[10] Misra, S., Myneni, S. (2010). On identifying power control performing Sybil nodes in wireless sensor networks using RSSI. In Global Telecommunications Conference (GLOBECOM 2010), IEEE, Miami, FL, USA, 6–10 Dec., pp. 1–5.

[11] Chen, S., Yang, G., Chen, S. (2010). A security routing mechanism against Sybil attack for wireless sensor networks. International Conference on Communications and Mobile Computing (CMC), IEEE, Shenzhen, China, 12–14 April, pp. 142–146.

[12] Jamshidi, M., Zangeneh, E., Esnaashari, M., Darwesh, A.M. and Meybodi, M.R. (2018). A Novel Model of Sybil Attack in Cluster-Based Wireless Sensor Networks and Propose a Distributed Algorithm to Defend It. *Wireless Personal Communications*, pp. 1–29 (in press).

[13] Zhang, Y., Fan, K.F., Zhang, S.B., Mo, W. 2010. AOA based trust evaluation scheme for sybil attack detection in WSN. *Application Research of Computers*, 27(5), pp. 1847–1849.

[14] Saxena, S., Sejwar, V. (2014). Sybil Attack Detection and Analysis of Energy Consumption in Cluster Based Sensor Networks. *International Journal of Grid and Distributed Computing*, 7(5), pp. 15–30.

[15] Ssu, K.F., Wang, W.T., Chang, W.C. (2009). Detecting Sybil attacks in Wireless Sensor Networks using neighboring information. *Computer Networks*, 53(18), pp. 3042–3056.

[16] Rafeh, R., Khodadadi, M. (2014). Detecting Sybil Nodes in Wireless Sensor Networks Using Two-hop Messages. *Indian Journal of Science and Technology*, 7 (9), pp 1359–1368.

[17] Muraleedharan, R., Ye, X., Osadciw, L.A. (2008). Prediction of Sybil attack on WSN using Bayesian network and swarm intelligence. In SPIE Defense and Security Symposium, International Society for Optics and Photonics, April 2008, pp. 69800F–69800F.

[18] Dhamodharan, U.S., Vayanaperumal, R. (2015). Detecting and Preventing Sybil Attacks in Wireless Sensor Networks Using Message Authentication and Passing Method. *The Scientific World Journal*, 1(1), pp. 13–17.

[19] Amuthavalli, R., Bhuvaneswaran, R.S. (2014). Detection and Prevention of Sybil Attack in Wireless Sensor Network Employing Random Password Comparison Method. *Journal of Theoretical & Applied Information Technology*, 67(1), pp. 236–246.

[20] Sarigiannidis, P., Karapistoli, E., & Economides, A. A. (2015). Detecting Sybil attacks in wireless sensor networks using UWB ranging-based information. *Expert Systems with Applications*, 42(21), pp. 7560–7572.

[21] Hu, R.H., Dong, X.M., Wang, D.L. (2015). Defense mechanism against node replication attacks and Sybil attacks in wireless sensor networks. *Acta Electronica Sinica*, 43(4), pp. 744–752.

[22] Jan, M.A., Nanda, P., He, X., Liu, R.P. (2016). A Sybil attack detection scheme for a forest wildfire monitoring application. *Future Generation Computer Systems*, 80, pp. 613–626.

[23] Jamshidi, M., Ranjbari, M., Esnaashari, M., Qader, N.N. and Meybodi, M.R. (2018). Sybil Node Detection in Mobile Wireless Sensor Networks Using Observer Nodes. *JOIV: International Journal on Informatics Visualization*, 2(3), pp. 159–165.

[24] Almas Shehni, R., Faez, K., Eshghi, F., Kelarestaghi, M. (2017). A New Lightweight Watchdog-Based Algorithm for Detecting Sybil Nodes in Mobile WSNs. *Future Internet*, 10(1), pp. 1–17.

[25] Jamshidi, M., Zangeneh, E., Esnaashari, M., Meybodi, M. R. (2017). A lightweight algorithm for detecting mobile Sybil nodes in mobile wireless sensor networks. *Computers & Electrical Engineering*, 64, pp. 220–232.

[26] Jamshidi. M., Darwesh, A.M., Lorenc, A., Ranjbari, M., Meybodi, M.R. (2018). A Precise Algorithm for Detecting Malicious Sybil Nodes in Mobile Wireless Sensor Networks. *IEIE Transactions on Smart Processing and Computing*, 7(6), pp. 457–466.

[27] Sobeih, A., Hou, J.C., Kung, L.C., Li, N., Zhang, H., Chen, W.P., Tyan, H.Y., Lim, H. (2006). J-Sim: a simulation and emulation environment for wireless sensor networks. *IEEE Wireless Communications*, 13(4), pp. 104–119.