IEIE Transactions on Smart Processing and Computing, vol. 7, no. 6, December 2018
https://doi.org/10.5573/IEIESPC.2018.7.6.457

A Precise Algorithm for Detecting Malicious Sybil Nodes in Mobile Wireless Sensor Networks

Mojtaba Jamshidi[1], Aso Mohammad Darwesh[2], Augustyn Lorenc[3], Milad Ranjbari[4], and Mohammad Reza Meybodi[5]

[1] Department of Information Technology, University of Human Development, Sulaimani, Iraq jamshidi.mojtaba@gmail.com
[2] Department of Information Technology, University of Human Development, Sulaimani, Iraq aso.darwesh@uhd.edu.iq
[3] Cracow University of Technology, Division of Logistics Systems, Poland alorenc@pk.edu.pl
[4] Department of Computer Engineering, Arak Branch, Islamic Azad University, Arak, Iran miladranjbari@gmail.com
[5] Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran mmeybodi@aut.ac.ir

* Corresponding Author: Mojtaba Jamshidi

Received October 12, 2018; Revised December 17, 2018; Accepted December 19, 2018; Published December 30, 2018

** Regular Paper*

Abstract: A Sybil attack, where a malicious node creates multiple fake or captured identities, is one of the most well-known attacks against wireless sensor networks (WSNs). This attack can leave devastating effects on operational and routing protocols, such as voting, data aggregation, resource allocation, and misbehavior detection. In this paper, a simple and precise algorithm for detecting Sybil attacks in mobile WSNs is proposed. Considering the rapid growth of Internet of Things (IoTs) devices and WSNs' popularity, the threat from this attack is serious. The main underlying idea of the proposed algorithm is to use neighbors' information and observer nodes to detect Sybil nodes during the network lifetime. In the proposed algorithm, some observer nodes first walk the network and record necessary information about other nodes. Each observer node then uses this collected information to detect Sybil nodes. The proposed algorithm is compared with other algorithms according to criteria including memory, communication, and computation overhead. Also, the proposed algorithm is implemented with the J-SIM simulator, and its performance is compared in a series of experiments with other algorithms using the criteria of true- and false-detection rates. The simulation results indicate that the proposed algorithm can detect 100% of the Sybil nodes, so its false-detection rate is 0%, regarding the study assumptions.

Keywords: Mobile sensor networks, Sybil attack, Lightweight algorithm, Observer node

1. Introduction

The rapid growth of technology enables us to develop sensors to collect data from many distributed sources. In transport and logistics, Internet of Things (IoTs) solutions are more common. Many companies invest in automatic solutions for internal transport and to control existing systems. Nowadays, the application of wireless sensor networks (WSNs) in the military, the environment, and for internal transport, traffic control, and surveillance operations signifies the importance of these networks. Given the unattended nature of WSNs and wireless communications, their broadcast nature, as well as network memory, processing, and energy constraints, there are a lot of attacks against wireless sensor networks [1, 2].

The Sybil attack, where a malicious node creates multiple fake or captured identities [3], is one of the most well-known attacks against these networks. This attack can leave devastating effects on operational and routing protocols such as voting, data aggregation, resource allocation, and misbehavior detection. An attack on WSNs in a logistics center or a production company can have tragic consequences. Taking control of the sensor network can expose the company to huge costs and can damage goods, stop a production line, or have a major impact, including loss of the brand's reputation [3-5].

Until now, many algorithms to counter the Sybil attack in fixed sensor networks have been developed, benefiting from mechanisms like random key pre-distribution [6], radio source tests [6], received signal strength indicator (RSSI) [7-9], trust evaluation [10], angle of arrival (AOA) identification [11], computational puzzles [12], and clustering [13-17]. In addition, some algorithms use

neighboring and routing information to detect Sybil nodes [18-20]. However, to detect mobile Sybil nodes, none of these mechanisms is applicable to mobile sensor networks, due to node mobility.

Algorithms have been presented to detect Sybil nodes in mobile sensor networks [21-24]. For a simulation in one paper, an algorithm was chosen as a base algorithm that uses observer nodes to detect Sybil nodes [24] based on the underlying idea of counting the number of sensor nodes facing the observer nodes, creating separate sets, and intersecting them. The base algorithm includes two phases: (1) each observer node stores the occurrences facing other sensor nodes in its history for R rounds, and (2) each observer node u navigates its history and generates distinct sets of node *IDs* such that each set includes the *IDs* of nodes that appeared an equal number of times in node u's neighborhood. Subsequently, the observer node selects the set where members are larger or equal to a threshold, T_{in}, and compiles a list of suspected Sybil nodes. In order to increase detection accuracy, observer nodes cooperate to detect Sybil nodes. More specifically, observer nodes send their suspected Sybil lists to each other. Each observer node creates a new list by intersecting all suspected Sybil lists.

The base algorithm encountered three main issues, despite its relatively high detection rate:

- Low convergence speed; i.e. the algorithm must run many times to achieve a favorable result.
- Observer nodes are not independent in the detecting process; i.e. they detect Sybil nodes by exchanging messages (which causes communications overhead and high memory usage).
- Overhead imposes many calculations on the observer nodes in order to create separate sets and to intersect them.

In this paper, we propose a simple and precise algorithm to detect Sybil nodes in mobile WSNs and to overcome the limitations of the base algorithm. The proposed algorithm uses neighbors' information and observer nodes to detect Sybil nodes in mobile WSNs. The proposed algorithm consists of two phases. In the first phase, the observer nodes monitor the mobility of other nodes for R rounds of network activity. In each round of network activity, each observer node discovers its current neighbors and updates its history if the number of neighbors exceeds a set threshold. In the second phase, each observer node decides on the originality of the other nodes using the information collected during the first phase.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 explains system assumptions and the attack model. Section 4 elaborates on the proposed algorithm. Section 5 explains the simulation results, and the last section concludes the paper.

2. Related Work

The Sybil attack was first introduced when Douceur [3] noted that peer-to-peer networks are vulnerable to this attack. Karlof and Wagner stated that the attack can affect routing protocols of sensor networks [5]. First, Newsome et al. [6] systematically analyzed a Sybil attack on wireless sensor networks, and introduced mechanisms like key pre-distribution, the radio source test, identity registration, and remote authentication code to deal with the attack. In the radio source test technique, each node assigns a separate signaling channel to each of its neighbors. This technique cannot be efficient, considering sensor node limitations. The identifier registration mechanism uses a central validation management unit and voting in the network. Generally, protective mechanisms based on voting or reputation approaches cannot work effectively because some nodes are usurpers and cannot be trusted to provide reliable information. In addition, identity authentication techniques require a huge memory space to save essential identity authentication information (such as shared encryption keys and identity certificates) and get involved in processing complex check algorithms.

In [7], an RSSI-based locating algorithm was proposed that uses the RSSI proportion of several receivers to estimate the location of nodes in a network. In [8] and [9], the locating mechanism proposed in [7] was used to detect Sybil nodes. The algorithm in [8] uses four location-aware nodes (tracking nodes) capable of hearing packets throughout the network. Tracking nodes cooperate to locate any nodes sending packets. This is sufficient to detect Sybil nodes, since all of them are positioned in nearby locations. RSSI-based algorithms also cannot be an appropriate solution, since radio signals are prone to interference from the environment and, as a result, the detection precision of such algorithms is affected. In [10], another strategy for detecting a Sybil attack was proposed based on time difference of arrival (TDOA) between the source and beacon nodes, which identifies the Sybil attack as well as the Sybil node location.

In [11], trust evaluation based on AOA (TEBA) was proposed. Taking into account that a Sybil node can create multiple identifiers, but has only one physical location, nodes are considered Sybil nodes when signal phase differences are less than a trust threshold (which is calculated by evaluating trust degrees for neighboring sensor nodes). In [12], a client puzzles–based algorithm was proposed to detect Sybil nodes in wireless sensor networks. The main idea with these algorithms is to prevent malicious nodes from solving the puzzles, created per the Sybil identities, at the right time. Other algorithms were proposed for detecting Sybil nodes in cluster-based sensor networks [13-17]. The algorithm proposed in [18] uses the concept of common neighbors to detect Sybil nodes. In [19], an algorithm was proposed for detecting a Sybil attack on multicast routing protocols based on geographic location. In [20], a method was developed that collects route information using a swarm intelligence algorithm during network operation, which detects Sybil nodes through their energy changes in the course of network activity.

In [21], a centralized algorithm was proposed that includes three phases: clustering, selecting nodes in the vicinity of a Sybil node, and routing procedures. So, it

cannot be a proper algorithm. A previous algorithm [22] was proposed to detect Sybil nodes in mobile wireless sensor networks. This algorithm employs mobile sink nodes and uses a mechanism to generate random numbers, exchanging them between the sink and sensor nodes to detect Sybil nodes. This algorithm is only applicable in a situation where a malicious node propagates some legal identities as its Sybil nodes (a stolen-identities model [6]). Another previous algorithm uses a distributed labeling mechanism to assign a bit label to nodes based on their movement [23]. This algorithm requires exchanging many messages between watchdog nodes, which increases communications overhead and power consumption as a result. Added to this, the algorithm has a relatively low Sybil node detection rate.

3. System Assumptions, the Attack Model, and symbols

The network in this study contains two sets of nodes (observer nodes and normal sensor nodes). Normal sensor nodes perform network missions, and observer nodes are responsible for detecting Sybil nodes. All sensor nodes (normal and observer) are mobile and, over the lifetime of the network, perform actions according to mobility models, such as random waypoint. Each node has a unique identity and ignores its location. The nodes communicate via wireless radio channels (of equal radio frequency r) and use an omnidirectional distribution approach. It is assumed that the observer nodes are equipped with tamper-resistant hardware, and that enemies cannot decode and reprogram them [24]. However, it is also assumed that all nodes have the same capabilities in terms of battery, memory capacity, and processing power).

Given the mobility of sensor nodes in the network environment, nodes must periodically send hello, routing, data, and sustain messages (after each period t or after they arrive at a new location in the network). In fact, that is a requirement for mobile sensor networks, so that each node can instantly identify its neighbors, set up a security key with them (if necessary), communicate, establish their own routing tables, and so on. [21-24]. Observer nodes stop sending such messages periodically, so they remain hidden, because they are responsible for detecting malicious nodes (i.e. Sybil nodes).

The attack model considered here is taken from other developed models [6, 18, 24]. With regard to the classification of the attack model in [6], our selected model is based on direct, simultaneous, and fake or fabricated identities. As shown in [18] and [24], an enemy can set up a Sybil attack and disrupt network function in the two ways presented above.

First, an enemy captures multiple normal nodes in the network, reprograms them as malicious nodes, and spreads them throughout the network again so that each malicious node presents multiple Sybil identities (two or three). Security algorithms can hardly ever identify Sybil nodes under such conditions, and some algorithms (such as the one proposed in [18]) may not even be able to identify any Sybil nodes. However, it is both difficult and time consuming for the enemy to capture, decode, reprogram, and control several normal nodes.

In the second form of attack, the enemy captures fewer normal nodes to be reprogrammed as malicious nodes. So, each malicious node exhibits more Sybil identities. In the second form, the enemy can disrupt many more routing protocols and network operations. In the proposed algorithm, as in [18], it is assumed that the enemy selects the second form.

It is also assumed that malicious nodes, as normal sensor nodes in the network environment, are mobile. Also, it is assumed the malicious nodes are located within network coverage. In addition, each malicious node must send hello and route request, etc., messages for each of its own Sybil identities after arriving at a new location in the network (the simultaneous Sybil attack model [6]). Indeed, this last assumption is necessary for the effectiveness of an attack by the enemy. If a malicious node does not expose all of its Sybil identities at the same time, it cannot disrupt routing protocols or affect network operations (such as voting, resource allocation, etc.).

Before introducing the proposed algorithm, it is pertinent to define the corresponding notations, as follows:

- n: the total number of nodes in the network.
- *history*: a table of observer nodes in memory.
- R: the number of iterations for network traffic monitoring by observer nodes (the number of iterations in the first phase of the proposed algorithm).
- α: a security parameter for the proposed algorithm.
- T_h: a threshold to determine which neighbors to suspect as Sybil nodes.
- P_t: a threshold to determine which nodes are Sybil nodes.
- d: the average number of neighbors to a node in the network.
- DN: the number of observer nodes in the network.
- SN: the number of normal sensor nodes in the network.
- M: the number of malicious nodes in the network.
- S: the number of Sybil *IDs* propagated by each malicious node.
- r: the radio range of nodes.

4. The Proposed Algorithm

The main idea of the proposed algorithm is based on three issues. First, all the Sybil nodes belong to only one malicious hardware node; therefore, they move simultaneously. Second, the existence of a malicious node that launches the Sybil attack at a specific location of the network increases node density at that point in the network. Third, all Sybil nodes attached to a malicious node appear equally in the neighborhood of other nodes (for example, observer nodes).

In a network, the proposed algorithm runs only in a limited number of nodes, called observer nodes, in a fully independent way. It is obvious that observer nodes contain

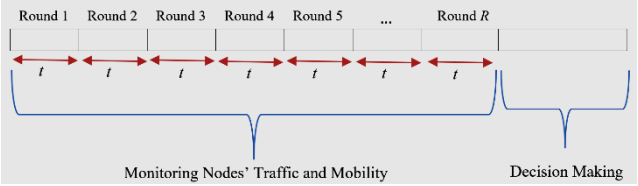

Fig. 1. The time line of the proposed algorithm.

NodeID	*Count*
*N*1	0
*N*2	0
*N*i	0
*N*n	0

Fig. 2. The structure of the history matrix for the observer nodes.

other algorithms, as well, to counteract other attacks. The proposed algorithm is a very simple, lightweight one that makes it a suitable and applicable algorithm for sensor networks. As shown in Fig. 1, the proposed algorithm contains two simple phases: (1) monitoring the nodes' traffic and mobility, and (2) decision making. A detailed explanation of the two phases is given below.

4.1 Monitoring Node Traffic and Mobility

Observer nodes regularly collect information required for detecting possible Sybil nodes in the network and save it in a matrix called a *history*. Each observer node has a *history*, which can be represented in memory by the matrix shown in Fig. 2 and which contains n rows (the total number of nodes in the network). The field *NodeID* shows the sensor node's identity, and the field *Count* stores an integer number. The initial value of *Count* is zero for all entries.

Observer nodes run the first phase of the proposed algorithm for R rounds at time intervals of t, as shown in Fig. 1. In each round of the monitoring phase, each observer node monitors its neighboring traffic for time slice t and stores the *ID* of current neighbors in a temporary vector, called *current_neighbor*. Each observer node discovers its current neighbors from the hello, routing request, etc., messages sent by sensor nodes. Then, if the number of current neighbors is more than a threshold value ($T_h = \alpha \times d$), the observer node (suspicious of existing Sybil nodes) increments the *Count* field of all current neighbors in its *history* by 1. Here, d is the average number of neighbors of a node, which can be calculated with environment area ($X \times Y$), the total number of nodes (n), and the radio range of the nodes (r), as seen in Eq. (1) [21]:

$$d = (\frac{n}{X \times Y} \times r^2 \times \pi) - 1 \quad (1)$$

It should be noted that the existence of a Sybil attack in region L in the network causes an average number of neighbors greater than d [21]. Therefore, the security parameter α must be greater than 1. But by increasing α, the detection rate of Sybil nodes will decrease, because observing nodes are less suspicious that Sybil nodes exist. Experiment 3 deals with this problem in detail. In each run of the monitoring phase, if the number of existing nodes in the neighboring area of an observer node is much more than usual (more than T_h), the observer node may consider them Sybil nodes. A Sybil attack in the network is one of the definite causes of unusual increases in the number of neighbors.

At the end of each time slice t, each observer node removes its *current_neighbor* vector and moves to another location in the network. After arriving at a new location in the network, the observer nodes launch the next run of the monitoring phase. The observer nodes independently repeat this process R times during the network lifetime and record the required data in their *history*. Then, the next phase, decision making, will be launched.

Editor - Highlight – As explained above, if this is the spelling used for this field name, please leave it unchanged.

4.2 Decision Making

Each observer node independently examines its *history* to identify Sybil nodes, if present. The decision-making process is simple. Each observer node scans its *history* and if, per each node u, the value of the field *Count* is $\geq P_t$, the observer node marks it as a Sybil node. P_t can be calculated with Eq. (2):

$$P_t = R \times (\frac{1}{n-1} \times d) \quad (2)$$

where R is the number of times the monitoring phase is run, n is the total number of nodes in the network, and d is the average number of neighbors for each node, which is calculated with Eq. (1). The reason for this decision is very clear: assuming there is a malicious node in the network, the possibility of discovering a malicious node in a single run of the monitoring phase, as a neighbor of an observer node, is $(\frac{1}{n-1} \times d)$, and in R runs, it equals P_t.

Also, since in each encounter of a malicious node with an observer node, the number of neighbors increases to more than the threshold (T_h), and a unit will be added to the field *Count* in the observer node's *history*, after R runs, the field *Count* corresponding to Sybil nodes will be $\geq P_t$.

Choosing the number of observer nodes required in the proposed algorithm can be a challenge. The dynamic nature of the mobile sensor network has made it impossible to determine the exact number of observer nodes needed in the network. Certainly, the minimum number of required observer nodes is just one. But, the maximum number of required observer nodes can be estimated in an ideal case in which all observer nodes cover the whole network environment together. Hence, the maximum number of required observer nodes can be calculated with Eq. (3):

$$\max(DN) = \left\lceil \frac{X \times Y}{r^2 \times \pi} \right\rceil \quad (3)$$

5. Performance Evaluation and Simulation Results

In this section, we first evaluate the overhead of the proposed algorithm in terms of memory, communication, and computation overhead. Then, we simulate our proposed algorithm and evaluate its performance through experiments.

5.1 Performance Evaluation

Memory Overhead: Since the proposed algorithm is implemented only by observer nodes, it does not impose any memory overhead on normal nodes. Each observer node requires $2n$ in memory to store its history matrix. It is assumed that each row of the *history* matrix occupies six bytes (four bytes for *NodeID* and two bytes for *Count*). Therefore, the memory overhead of the proposed algorithm follows is $O(n)$ bytes per observer node, and is $O(DN \times n)$ bytes, per the proposed algorithm. Memory overhead of the base algorithm in [24] is $O(DN^2 \times n)$ bytes, where DN is the number of observer nodes.

Communication Overhead: The proposed algorithm does not impose any communication overhead on normal sensor nodes and observer nodes because neither one sends a message when the algorithm is running; but, observer nodes only use the hello messages propagated by sensor nodes. Periodic communication of hello messages is a requirement in mobile sensor networks. However, the communication overhead of the base algorithm in [24] equals $(DN \times (DN-1) \times k)$, where k is the network diameter.

It should be noted that sending, receiving, and processing packets are the main energy-consuming operations in sensor networks. Sending packets consumes much more energy than processing and receiving packets; hence, the communication overhead determines the amount of energy consumed by an algorithm. Since the proposed algorithm does not impose any communication overhead on observer nodes, it is an energy-efficient algorithm, especially in comparison to the base algorithm in [24].

Computation Overhead: Neither the proposed algorithm nor the base algorithm [24], imposes computation overhead on normal sensor nodes, but does on observer nodes. In the proposed algorithm, during each run of the monitoring phase, if the number of neighbors exceeds the threshold (T_h), one unit will be added to the field *Count*. So, after the first phase, computation overhead with the proposed algorithm per observer node is $O(R \times d)$. Yet, computation overhead of the proposed algorithm in the decision-making phase is $O(n)$ because all rows of the *history* matrix and all rows with $Count \geq P_t$ have to be marked as Sybil nodes. But, the computation overhead of the base algorithm in [24] is $O(m^2 \times DN)$ (m is the average number of distinct sets for each observer node).

Comparisons of the overhead in the proposed algorithm and the base algorithm are in Table 1. Performance evaluation shows that the proposed algorithm imposes low computation, memory, and communication overhead on observe nodes, and no overhead on other sensor nodes. Therefore, the proposed algorithm is lightweight and applicable in real environments.

Table 1. Performance comparison of the proposed algorithm and the base algorithm in terms of memory, communication, and computation overhead per observer node.

Algorithm	Memory overhead	Communication overhead	Computation overhead
[24]	$O(DN \times n)$	$O(DN \times k)$	$O(m^2 \times DN)$
Proposed algorithm	$O(n)$	0	$O(R \times d + n)$

5.2 Simulation Model

The proposed algorithm was implemented with the J-SIM simulator [25], and its performance was compared through a series of experiments with the base algorithm [24] in terms of (1) true-detection rate, i.e. the percentage of detected Sybil nodes, and (2) false-detection rate, i.e. the percentage of normal nodes erroneously identified as Sybil nodes.

The considered simulation model was adapted from the base algorithm in [24], and therefore, it is assumed that the network contains n=300 sensor nodes distributed in a 100 m × 100 m area. Also, it is assumed that the enemy has captured M=5 legitimate nodes in the network and reprogrammed them as malicious nodes, so that each malicious node propagates S fake identities (Sybil nodes) after being distributed in the network.

Also, DN nodes of the network are regarded as observer nodes. All the nodes (legitimate and malicious) have the same radio range (r=10 meters). What is more, the random mobility model for previous research [23, 24, 26] was used for node movement. The monitoring phase of both the proposed algorithm and the base algorithm was run R times. In the base algorithm, T_{min}=10. Each experiment was repeated 100 times, and the final result is the average of these 100 times.

5.3 Experiment Results

Experiment 1: In this experiment, the effect of parameter R on the performance of the proposed algorithm is evaluated, and the results are compared with the base algorithm. The considered values here are DN=5 and 10, and $\alpha = 1.6$. Fig. 3 indicates the true-detection rate, and Figs. 4 and 5 indicate the experiment results in terms of false-detection rate.

The experiment results in Fig. 3 show that by increasing the number of observer nodes in the network, the true-detection rate increases in both algorithms because more network areas are simultaneously monitored by observer nodes. Increasing the number of observer nodes slightly increases the false-detection rate with the proposed algorithm, whereas it reduces this metric in the base algorithm. This is because, in the proposed algorithm, observer nodes detect Sybil nodes in a fully independent

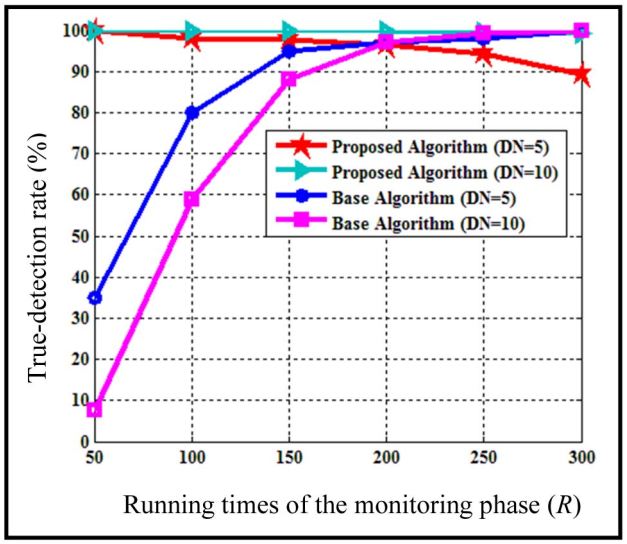

Fig. 3. Effect of parameter *R* on the true-detection rate: comparing the proposed algorithm with the base algorithm.

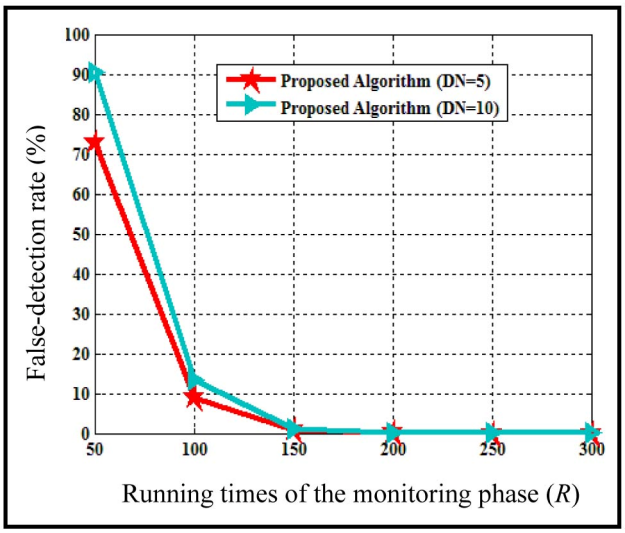

Fig. 4. Effect of parameter *R* on the false-detection rate: the proposed algorithm.

way. Therefore, increasing the number of observer nodes slightly increases its false-detection rate. But, in the base algorithm, observer nodes cooperate and do intersect to identify Sybil nodes, which decreases its false-detection rate. However, as indicated in Figs. 4 and 5, after R=100, the false-detection rate for the proposed algorithm is about 10%, which is almost 0% if we assume R>150. On the other hand, the false-detection rate of the base algorithm, after R=100 is almost 30%, and even after repeating the experiment for R=600, it would be about 3%.

Also, the experiment results presented in Fig. 3 indicate that the true-detection rate of the proposed algorithm is always more than the base algorithm for DN=10. If DN=10, the proposed algorithm performs better than the base algorithm at $R \le 200$; but at $R > 200$, the true-detection rate of the proposed algorithm is reduced. By increasing the running times of the monitoring phase, the possibility

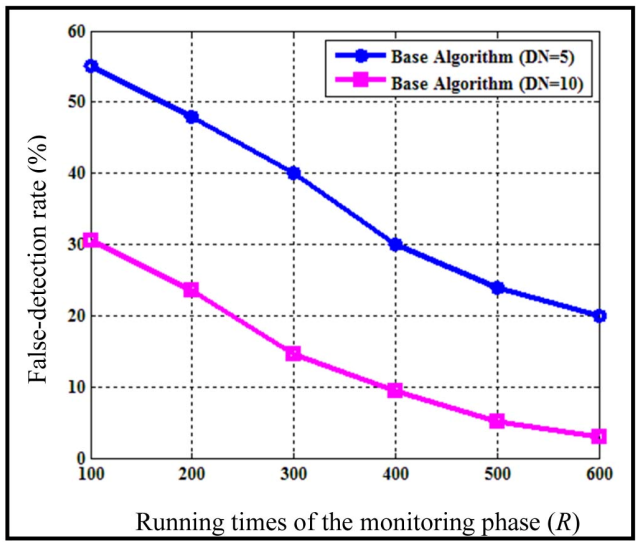

Fig. 5. Effect of parameter *R* on the false-detection rate: base algorithm.

that a malicious node appears at least P_t times in the neighboring environment of the observer nodes decreases, which results in a reduced true-detection rate. Although increasing the number of observer nodes will overcome this, as the experiment results clearly show, when R=150, the proposed algorithm gives favorable results, and there is no need to repeat the monitoring phase any more (the detection rate is about 100% and the false-detection rate is about 0%).

The experiment results prove the performance of the proposed algorithm is better than the base algorithm in terms of both true-detection and false-detection rates. Also, it is found that the proposed algorithm clearly performs better than the base algorithm, considering the false-detection rate. Thus, in the following experiments, no comparison is made between the false-detection rates of these two algorithms.

Experiment 2: The purpose of this experiment is to examine the effect of the number of Sybil nodes (S), propagated by every malicious node, on the true and false-detection rates of the proposed algorithm in comparison with the base algorithm. In this experiment, we have DN=10, $\alpha = 1.6$, and the number of Sybil identities propagated by each malicious node at S=10 and 15. Fig. 6 indicates the experiment results in terms of the true-detection rate, and Fig. 7 indicates the experiment results in terms of false-detection rate.

The experiment results in Fig. 6 show that by increasing S, the true-detection rate increases with the proposed algorithm. The more we increase the number of Sybil nodes, S, the greater the possibility of *having* more than the threshold neighbors (T_h) around the observer node (that is, of course, when there is a malicious node in the observer node's neighborhood). Thus, by increasing the corresponding field *Count*, we have the required condition of having a Sybil node from the decision-making phase. Also, the experiment results in Fig. 7 indicate that changes in parameter S, can slightly affect the false-detection rate of the proposed algorithm. But if the number of Sybil

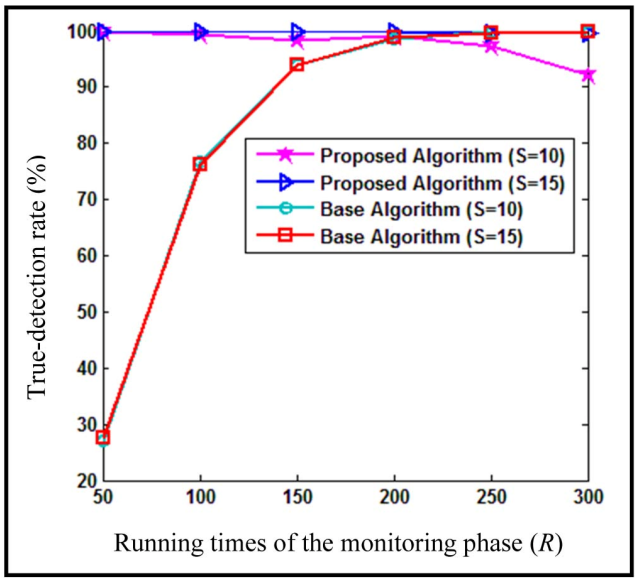

Fig. 6. Effect of parameter *S* on the true-detection rate: Comparing the proposed algorithm with the base algorithm.

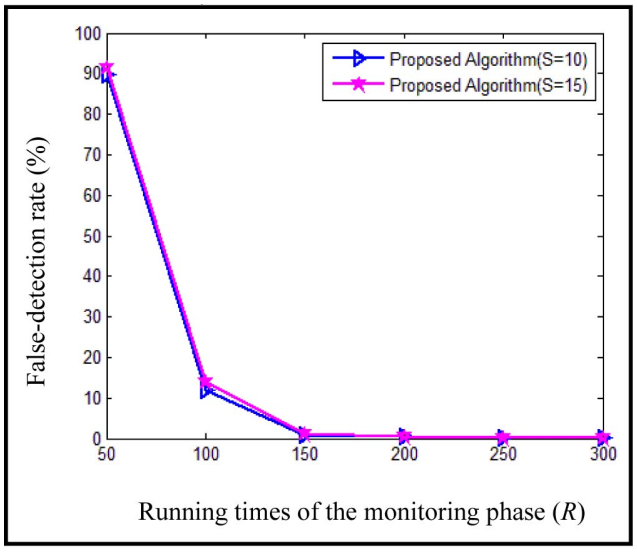

Fig. 7. Effect of parameter *S* on the false-detection rate: the proposed algorithm.

identities is too low (only two or three), the proposed algorithm's performance, like other algorithms that use neighbor information to detect Sybil nodes, will decrease drastically. As mentioned before, no enemy can leave many devastating effects on a network with only a small number of Sybil nodes.

Experiment 3: The purpose of this experiment is to examine the effect of α on the performance of the proposed algorithm. In this experiment, we have DN=10, S=20, and α = 1~1.8 . Fig. 8 indicates the experiment results in terms of true-detection rate, and Fig. 9 indicates the experiment results in terms of false-detection rate.

The experiment results show that with an increasing α, the true-detection rate and the false-detection rate decrease with the proposed algorithm. This is because, with an

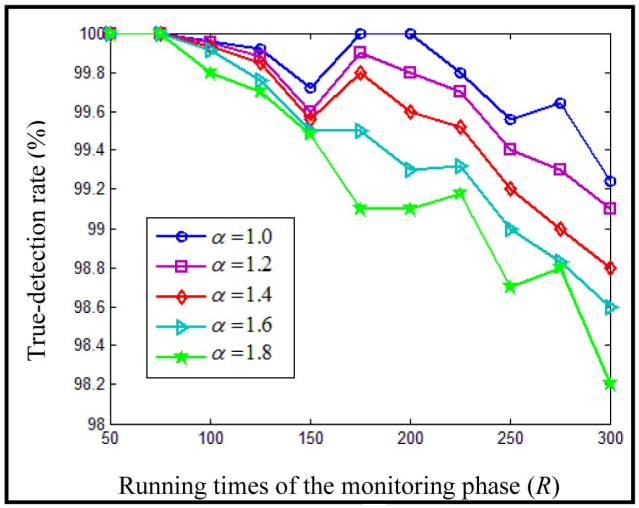

Fig. 8. Effect of parameter α on the true-detection rate: the proposed algorithm.

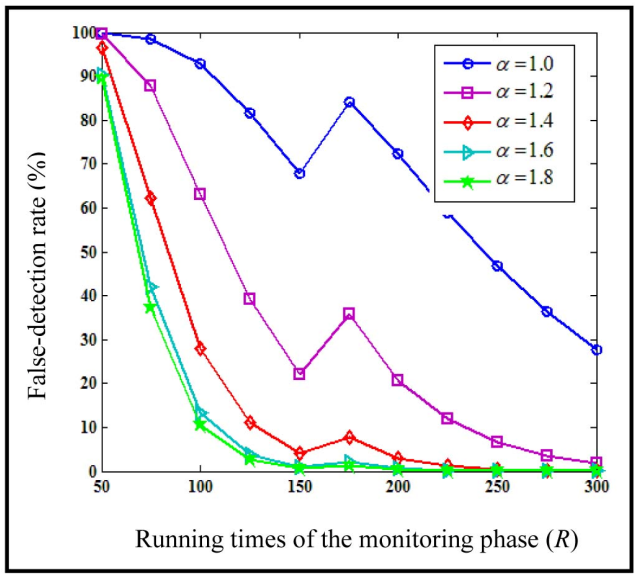

Fig. 9. Effect of parameter α on the false-detection rate: the proposed algorithm.

increasing α , T_h also increases, and the possibility of having more neighbors than the threshold (T_h) around the observer node decreases (even if there are Sybil nodes in the neighboring area). This leads to a decrease in the true-detection rate under the proposed algorithm. But as Fig. 9 indicates, the amount of α significantly affects the false-detection rate. With a decreasing α , T_h also decreases, and this causes the number of nodes that are present in the neighborhood of the observer nodes to be more than T_h, even under conditions where there are no Sybil nodes in the network. The result increases the false-detection rate of the proposed algorithm.

Experiment 4: In this experiment, the true- and false-detection rates of the proposed algorithm, on average, are compared with other algorithms. Table 2 shows the results of this comparison. As shown, the proposed algorithm with an average true-detection rate of 99%, falls below the

Table 2. Performance comparison of the proposed algorithm and other algorithms in terms of average false- and true-detection rates.

Algorithm	Description of algorithm	Average true-detection rate	Average false-detection rate
[8]	Uses RSSI and four tracking nodes that can listen to packets across the network to estimate the location of nodes (for static networks).	100%	6%
[9]	Uses RSSI to detect a Sybil attack, even if the Sybil nodes change their transmission power for each Sybil *ID* (for static networks).	98%	6%
[13]	Uses RSSI and clustering to detect Sybil nodes (for static clustered networks).	92%	2%
[14]		90%	1%
[18]	Uses neighboring information and discovers common neighbors to detect Sybil nodes (for static networks).	99%	5%
[23]	Uses a distributed labeling mechanism to assign a bit label to nodes based on their movement.	94%	0%
[24]	Uses observer nodes to count the number of sensor nodes facing observer nodes, and creates separate sets to detect Sybil nodes (for mobile networks).	99%	5%
The proposed algorithm	Uses neighbors' information and observer nodes to detect Sybil nodes (for mobile networks).	99%	0%

algorithm proposed in [8], but excels compared to the rest. Moreover, the proposed algorithm and algorithm [23] exceed the others in terms of average false-detection rate (0%). Some Sybil-detection algorithms apply only to fixed and non-mobile sensor networks [8, 9, 13, 14, 18]. These algorithms are not dynamic, and make a decision only once, so malicious nodes can be placed in a location that cannot be detected. Even some legal nodes are randomly placed near malicious nodes, and are thus mistakenly detected as malicious nodes or Sybil nodes. But in mobile-specific algorithms, such as the proposed algorithm, the detection process can operate after a monitoring phase that takes place in different locations and situations in the network. Therefore, they can identify Sybil nodes more accurately. In [8], the algorithm is based on RSSI, and its accuracy sharply drops in noisy environments.

6. Conclusion

The current study proposed a simple and precise algorithm using observer nodes to detect Sybil attacks in mobile wireless sensor networks. In doing so, the algorithm uses node movement information in the network environment as well as network density. The proposed algorithm contains two phases: monitoring and decision making. In the monitoring phase, observer nodes collect the data required to detect Sybil nodes in a decision-making phase using the collected data. The proposed algorithm's overhead in memory, communication, and processing was analyzed, and the results of the simulations proved its lightweight nature when compared with a base algorithm. In addition, to show the proposed algorithm's efficiency, it was first implemented and then its performance was compared in a series of experiments with the base algorithm in terms of true- and false-detection rates, and with other existing algorithms in terms of average true- and false-detection rates. The experiment results proved the more-favorable performance of the proposed algorithm.

References

[1] J. Yick, B. Mukherjee and D. Ghosal. 2008. Wireless sensor network survey. Computer Networks. 52(12): 2292–2330. Article (CrossRef Link)

[2] M. Jamshidi, A. A. Shaltooki, Z. D. Zadeh and A. M. Darwesh. 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). Article (CrossRef Link)

[3] J. R. Douceur. 2002. The Sybil attack. Proc. First International Workshop on Peer-to-Peer Systems (IPTPS '02): 251-260. Article (CrossRef Link)

[4] A. Diaz and P. Sanchez. 2016. Simulation of attacks for security in wireless sensor network. Sensors. 16(11): 1-27. Article (CrossRef Link)

[5] C. Karlof and D. Wagner. 2003. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. AdHoc Networks. 1(2): 299-302. Article (CrossRef Link)

[6] J. Newsome, E. Shi, D. Song and A. Perrig. 2004. The Sybil attack in sensor networks: analysis and defenses. Proc. International Symposium on Information Processing in Sensor Networks: 259–268. Article (CrossRef Link)

[7] S. Zhong, L. Li, Y. G. Liu and Y. R. Yang. 2004. Privacy-preserving location based services for mobile users in Wireless Networks. Yale University, Dept. of Comp. Sci. Technical Report ALEU/DCS/TR-1297. Article (CrossRef Link)

[8] M. Demirbas and Y. Song. 2006. An RSSI-based scheme for Sybil attack detection in wireless sensor networks. Proc. IEEE Computer Society International Symposium on World of Wireless, Mobile and Multimedia Networks: 570–574. Article (CrossRef Link)

[9] S. Misra and S. Myneni. 2010. On Identifying Power Control Performing Sybil Nodes in Wireless Sensor Networks Using RSSI. Proc. IEEE Global Telecommunications Conference (GLOBECOM): 1-4. Article (CrossRef Link)
[10] M. Wen, H. Li and Y.-F. Zheng. 2008. TDOA-based Sybil attack detection scheme for wireless sensor. Journal of Shanghai University. 12(1): 66-70. Article (CrossRef Link)
[11] Y. ZHANG, K.-F. FAN., S.-B. ZHANG and W. MO. 2010. AOA based trust evaluation scheme for Sybil attack detection in WSN. Appl. Res. Comput. 27(5): 1847-1849. Article (CrossRef Link)
[12] F. Li, P. Mittal, M. Caesar and N. Borisov. 2012. SybilControl: Practical Sybil Defense with Computational Puzzles. Proc. Seventh ACM workshop on Scalable trusted computing, ACM: 67-78. Article (CrossRef Link)
[13] S. Chen, G. Yang and S. Chen. 2010. A Security Routing Mechanism against Sybil Attack for Wireless Sensor Networks. Proc. IEEE International Conference on Communications and Mobile Computing: 142-146. Article (CrossRef Link)
[14] Q. Zhang, P. Wang, D. Reeves and P. Ning. 2005. Defending against Sybil attacks in sensor networks. Proc. Second International Workshop on Security in Distributed Computing Systems: 185–191. Article (CrossRef Link)
[15] M. Jamshidi, E. Zangeneh, M. Esnaashari, A. M. Darwesh and M. R. Meybodi. 2018. A Novel Model of Sybil Attack in Cluster-Based Wireless Sensor Networks and Propose a Distributed Algorithm to Defend It. Wireless Personal Communications: 1-29. Article (CrossRef Link)
[16] J. Yang, Y. Chen and W. Trappe. 2008. Detecting Sybil attack in wireless and sensor networks using cluster analysis. Proc. 5th IEEE international Conference on Mobile Ad hoc and Sensor Systems, Atlanta, GA: 834-839. Article (CrossRef Link)
[17] X.-D. WANG, Y.-Q. SUN and X.-X. MENG. 2009. Cluster-based Defending Mechanism for Sybil Attacks in Wireless Sensor Network. Computer Engineering. 15:129-131. Article (CrossRef Link)
[18] K. F. Ssu, W. T. Wang and W. C. Chang. 2009. Detecting Sybil attacks in wireless Sensor Networks using neighboring information. Computer Networks. 53(18): 3042–3056. Article (CrossRef Link)
[19] S. Ramachandran and V. Shanmugan. 2012. Impact of Sybil and Wormhole Attacks in Location based Geographic Multicast Routing Protocol for Wireless Sensor Networks. Journal of Computer Science. 7(7): 973-979. Article (CrossRef Link)
[20] R. Muraleedharan, X. Ye and L.A. Osadciw. 2008. Prediction of Sybil Attack on WSN Using Bayesian Network and Swarm Intelligence. Proc. SPIE Defense and Security Symposium: 69800F-69800F. Article (CrossRef Link)
[21] S. Sharmila and G. Umamaheswari. 2012. Detection of Sybil attack in Mobile Wireless Sensor Networks. Journal of engineering science & advanced technology. 2(2): 256 – 262. Article (CrossRef Link)
[22] A. Andalib, M. Jamshidi, F. Andalib and D. Momeni. 2016. A Lightweight Algorithm for Detecting Sybil Attack in Mobile Wireless Sensor Networks using Sink Nodes. International Journal of Computer Applications Technology and Research. 5(7): 433 – 438. Article (CrossRef Link)
[23] M. Jamshidi, E. Zangeneh, M. Esnaashari and M. R. Meybodi. 2017. A lightweight algorithm for detecting mobile Sybil nodes in mobile wireless sensor networks. Computers & Electrical Engineering. 64: 220-232. Article (CrossRef Link)
[24] M. Jamshidi, M. Ranjbari, M. Esnaashari, N. N. Qader and M. R. Meybodi. 2018. Sybil Node Detection in Mobile Wireless Sensor Networks Using Observer Nodes. JOIV: International Journal on Informatics Visualization, 2(3): 159-165. Article (CrossRef Link)
[25] J-SIM Simulator, https://sites.google.com/site/jsimofficial/, December 25, 2016. Article (CrossRef Link)
[26] C. M. Yu, C. S. Lu and S. Y. Kuo. 2008. Mobile Sensor Network Resilient Against Node Replication Attacks. Proc. IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON): 597-599. Article (CrossRef Link)

Mojtaba Jamshidi received the B.S. degree in Computer Engineering from the Iranian Academic Center for Education, Culture and research (ACECR), Kermanshah, Iran, in 2009, and M.S. degree in Computer Engineering from Islamic Azad University, Qazvin, Iran, in 2012. His research interests include computer networks, learning systems, security, meta-heuristic algorithms, data mining, and recommender systems.

Aso Mohammad Darwesh received the B.S. in Mathematics in University of Sulaimani, Iraq 2001, M.S. degrees in Computer Science in University of Rene Descartes, France 2007 and PhD in Computer Science, University of Pierre and Mari Curie, France 2010. Currently he is associate Professor in Information Technology Department, University of Human Development, Sulaimani, Iraq. His research interests include Serious Games, learning system, Computer Network, and Data mining.

Augustyn Lorenc works in Institute of Rail Vehicle at Cracow University of Technology. Since 2006, he cooperates with Warsaw advertisement agency JM Group as web application developer. In 2012, he was running his own IT company in the Academic Business Incubator. In 2016, he started work for SKK S.A. as R&D Engineer. His research interests include supply chain management, cargo forwarding and securing, artificial neural network, warehouse logistics, database design, on-line systems design and transport and logistics market. He is author/co-author of 60 scientific papers. He is also participant of many research projects for railway companies.

Milad Ranjbari received the B.S. degree in Computer Engineering from the Payame Noor University, Kermanshah, Iran, in 2010, and the M.S. degree in Computer Engineering from Islamic Azad University, Arak, Iran, in 2015. My research interests include cloud computing, wireless sensor network, and learning systems.

Mohammad Reza Meybodi received the B.S. and M.S. degrees in Economics from the Shahid Beheshti University in Iran, in 1973 and 1977, respectively. He also received the M.S. and Ph.D. degrees from the Oklahoma University, USA, in 1980 and 1983, respectively, in Computer Science. Currently he is a Full Professor in Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. Prior to current position, he worked from 1983 to 1985 as an Assistant Professor at the Western Michigan University, and from1985 to 1991 as an Associate Professor at the Ohio University, USA. His research interests include, channel management in cellular networks, learning systems, parallel algorithms, soft computing and software development.