Ad Hoc Networks 100 (2020) 102081

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

A simple, lightweight, and precise algorithm to defend against replica node attacks in mobile wireless networks using neighboring information

Mojtaba Jamshidi[a,*], Shokooh Sheikh Abooli Poor[b], Abbas Arghavani[c], Mehdi Esnaashari[d], Abdusalam Abdulla Shaltooki[e], Mohammad Reza Meybodi[f]

[a] Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
[b] Computer Engineering Department, Municipality Training Centre of Applied Science and Technology, Ahvaz, Iran
[c] Department of Computer Science, University of Otago, New Zealand
[d] Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran
[e] Department of Information Technology, University of Human Development, Sulaymaniyah, Iraq
[f] Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 6 June 2019
Revised 14 January 2020
Accepted 24 January 2020
Available online 29 January 2020

Keywords:
Wireless sensor network
Replica attack
Replica node
Security
Lightweight solution

ABSTRACT

Node replication attack (also known as replica attack) is one of the common attacks in Wireless Sensor Networks (WSNs). In this attack, the adversary captures the sensor node(s) and extracts all important information, properties, and functions. Then, it disrupts the network operation by generating several malicious nodes having the same ID(s) as that of the captured node(s). Since the malicious nodes (i.e., replica nodes) have valid keying material, they can intentionally interfere with the network operation. Motivated by this, a three-step mechanism is proposed to detect replica nodes. The proposed mechanism benefits from watchdog nodes and is based on the idea that all nodes in the network (which are identified by their unique IDs) should have the same chance to be met by watchdog nodes. Accordingly, watchdog nodes monitor the network traffic and silently overhear the channel to see if there is any node with an abnormally higher chance of being met. If so, the ID of the node is advertised as the replica node.

To evaluate the performance of the proposed mechanism in terms of probability of detecting replica nodes and false detection probability, a comprehensive set of simulations has been carried out using the J-SIM simulator. The simulation results reveal that the proposed mechanism detects almost all replica nodes while the false detection probability is below 0.005%. It outperforms state-of-the-art protocols with imposing ignorable processing and memory overhead.

1. Introduction

WSNs are a type of ad-hoc networks comprised of hundreds to thousands of small and cheap Sensor Nodes (SNs). They are widely applied in various applications from military and industry to even healthcare. They are mostly suitable for applications in which presence of humans (i.e., network operators) is costly, dangerous, or difficult, if not impossible. After deployment of nodes in the environment or upon completion of the mission, nodes cannot be collected and used again, therefore, the developing cost of SNs should be as low as possible. Considering low cost small size nodes, they face many constraints including memory capacity, computational power, radio range, energy, and etc. Considering these constraints and unattended deployment of SNs, wireless nature of communications, and ever-increasing applications of SNs in military, the security establishment in WSNs is an important and challenging issue which has attracted attention of many researchers [1–3].

One of the common attacks in WSNs is node replication attack or replica node. Considering unattended deployment of SNs in the operation environment, the adversary can capture some of the SNs and extract their important information including keying material to generate replica nodes. Since replica nodes include exact information and characteristic (including ID, keying material) of the captured SN, they can establish a key with other SNs (i.e., normal SNs). Then, the adversary distributes the replica nodes in the network, as presented in Fig. 1, and may establish other attacks.

* Corresponding author.
E-mail addresses: jamshidi.mojtaba@gmail.com (M. Jamshidi), shokooh_sheikh@yahoo.com (S.S.A. Poor), arghavani@cs.otago.ac.nz (A. Arghavani), esnaashari@kntu.ac.ir (M. Esnaashari), salam.abdulla@uhd.edu.iq (A.A. Shaltooki), mmeybodi@aut.ac.ir (M.R. Meybodi).

https://doi.org/10.1016/j.adhoc.2020.102081
1570-8705/

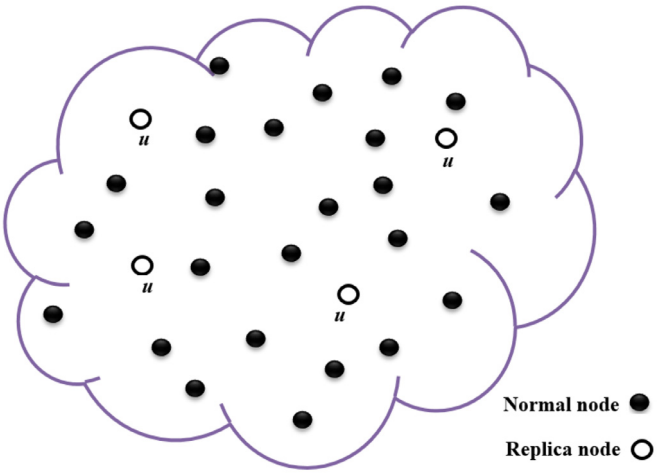


Fig. 1. An example of a replica node attack.

For instance, the adversary can easily monitor a large part of network traffic which passes through replica nodes, destroy the observation operation of the normal SNs by injecting distorted data and disrupt common protocols of the network including clustering and data aggregation. On one hand, the replica nodes are controlled by the adversary and on the other hand, they have ID and keying materials which makes them similar to normal SNs [4–6].

Although many algorithms have been proposed to detect replica nodes in WSNs [7–15], they cannot support mobile WSNs [16,17] because: (a) most of these algorithms rely on determining the location of SNs and transmitting location claims to witness nodes or specific locations in the network; (b) they are specific to particular topologies (like grid topology). However, some algorithms have been proposed to detect replica nodes in mobile WSNs [18–30], but they suffer from some disadvantages including communication overhead, high memory usage, non-scalability, complicated process of detecting replica nodes, the need for extra information (e.g., location information) extra hardware (e.g., GPS), or extra computationally expensive software modules (e.g., digital signatures and keying modules).

In this paper, a lightweight algorithm using watchdog nodes (WNs) [2,3] and neighboring information is proposed to detect replica nodes in mobile WSNs which resolves shortcomings of the existing algorithms. The proposed algorithm does not need the location of SNs, extra hardware, software, public keys, and digital signatures. The proposed algorithm operates only based on the number of the times that SNs appear in the neighborhood of the WNs. The rest of this paper is organized as follows. Section 2 reviews previous works. System assumptions and attack model are given in Section 3. Section 4 presents the proposed algorithm. The simulation based analysis of the efficiency of the proposed algorithm is presented in Section 5. Finally, the paper is concluded in Section 6.

2. Related work

In this section, we investigate the existing algorithms for detecting replica node attack in static and mobile WSNs.

2.1. Detection of replica attack in static WSNs

Line-Selected Multicast (LSM) [7] is a well-known distributed probability algorithm, which uses a public encryption key. In LSM, neighboring nodes guide each node to issue location claims towards some witness nodes, which are selected randomly. Each witness node or an intermediate node that receives more than one location claims for a particular node, like u, considers u as a replica node. In summary, the novelty of this algorithm is the use of message encryption along with spreading location claims from neighboring nodes towards some witness nodes, which are selected randomly. Disadvantages of LSM are high communication and memory overhead because nodes have to periodically multicast the location claims over the whole lifetime of the network.

Randomized, Efficient, and Distributed (RED) [8] is a centralized algorithm whose main idea is to transmit location claims (with digital signature) to different locations within the network, which are selected based on a random number issued by a central point. In summary, the novelty of RED is the use of a central point to broadcast a pseudorandom seed, ϑ, to all SNs periodically. Each neighbor node executes a publicly known hash function on ϑ to generate several pseudo-random witness locations, to which, location claims should be sent. Disadvantages of RED are high communication and memory overhead as well as the need for synchronization by a central point. In addition, the efficiency of RED in both sparse and dense networks may decrease significantly. In a sparse network, a case may arise where there will be only a few, if any, witness nodes within the selected witness location. In such a scenario, the claim may or may not reach a witness node, which reduces the detection probability. In a highly dense network, the average memory overhead of the network increases because the number of witness nodes in the selected witness location increases significantly.

RAndom WaLk (RAWL) and Table-assisted RAndom WaLk (TRAWL) [9] are proposed based on a random walk to detect replica nodes. In RAWL, for each node u, several random steps are taken, and nodes that have been passed are selected as witness nodes of u, and store the location claim. TRAWL algorithm is based on RAWL, but it adds a trace table to each node to reduce memory cost. Disadvantages of these algorithms are high communication and memory overhead; their overheads are even higher than that of RED and LSM.

Jamshidi et al. [10] proposed a novel algorithm based on a dynamic ID assignment mechanism to defend against replica node attacks in static WSNs. The novelty of this algorithm is the use of a multi-tree architecture based on the multi-sink architecture for dynamic assignment of ID to SNs after deployment in the network environment. If this mechanism is used, replica nodes generated by the adversary cannot be easily attached to the network. Disadvantages of this algorithm are twofold: (1) the need for having multiple sink nodes in the network; and (2) injecting new nodes to the network is not always easily applicable.

Yu et al. [11] proposed Compressed Sensing-based clone Identification (CSI) to detect replica nodes in the static WSNs. The key insight in designing CSI is that the number of replicas in a network is usually very limited. The main idea behind CSI is that each SN broadcasts a fixed number θ to its one-hop neighbors. This α can be considered as a pseudo sensor reading of each node. SNs aggregate and forward the received numbers from descendant nodes along an aggregation tree via compressed sensing-based data gathering techniques. The base station, as the root of the aggregation tree, performs compressed sensing recovery on the aggregated result. According to the reconstructed result, the base station marks the SN with a reading greater than θ as a replica node since a non-replica node can only report the number θ once. This algorithm is centralized and thus, is not scalable.

Mishra et al. [12] uses the deviation in the distance traveled by a SN and its replica is discovered by the observer nodes. Observer nodes keep a sliding window of the recent time-distance broadcast of the other nodes. A replica is detected by an observer node based on the degree of violation computed from the deviations recorded using the time-distance sliding window. A drawback of this algorithm is its low detection probability, especially in a sa-

tiation where an adversary injects a few number of clones into the network.

Ho et al. [13] proposed three detection schemes based on group deployment knowledge which adapts the location claim idea from [7] to defend against node replication attack. These schemes are based on the assumption that SNs are deployed in groups. In these schemes, SNs inside their home zone can communicate without any extra validation of security protocols, whereas SNs outside their home zone are not allowed to transmit messages unless they are authenticated by location claims. A drawback of this algorithm is that SNs in different groups cannot communicate together.

Ding et al. [14] used similarity estimation with a group deployment knowledge to detect clone nodes in WSNs. They prevent replicas from generating false location claims without deploying localization techniques on the SNs. Disadvantages of this algorithm are: (1) its high communication overhead and (2) it is applicable only in grid topologies.

Roy and Nene [15] proposed an algorithm based on Received Signal Strength Indicator (RSSI), Link Quality Indicator (LQI), and Packet Sequence Number (PSN) to detect replica nodes in WSNs. Also, they use a combination of broadcast and unicast messages. RSSI is used to obtain the amount of the residual battery life, LQI is dependent on the distance between the transmitter and the receiver nodes, and PSN is used to keep a check on any duplicate packet generated. Generally, using RSSI for determining location of SNs is not reliable because radio signal is prone to be interfered by the environment.

2.2. Algorithms in mobile WSNs

The main idea of the algorithm eXtremely Efficient Detection (XED) [18] is that if a node u meets another node v at time T_1, it sends a random number r to v at the same time. When nodes u and v meet each other once again at time T_2, u asks v for the random number which had sent to it at time T_1 and expected v to send back the same number r. If v is not a replica node, it returns r, but if it is a replica node, it might return another random number. The novelty of this algorithm is the use of a "remember and challenge strategy" to detect replica nodes. Disadvantages of this algorithm are (1) high memory overhead: each node requires a memory of size $3n$; (2) high communication overhead: in each round of movement, each node should acquire the random numbers from its neighbors. In addition, it should respond to its neighbors (send back their random numbers); and (3) security failure, if replica nodes cooperate and share their random table, the algorithm may fail.

Safari et al. [19] proposed Advanced XED (AXED) algorithm to combat XED's shortcomings. The AXED is like XED but in order to reduce the memory and communication overheads, each node has to observe only log n other nodes. The list of nodes being observed by each node u is selected before nodes deployment, randomly. Therefore, each node has to observe a few numbers of nodes and whenever two nodes meet, they ask for the previous random numbers first, then new random numbers are generated and exchanged. Employing this mechanism, means generating and exchanging new random numbers at each meeting, prevent algorithm from failure in the case of cooperating replica nodes. Drawbacks of this algorithm are: (1) it imposes communication and memory overheads on all nodes in the network (however, these overheads are lower than XED algorithm); (2) it has low detection probability and slow detection in sparse networks; and (3) injecting new normal nodes to the network is not always easily possible.

The main idea of algorithms presented in [20,21] is inspired by the fact that a normal node should not move faster than the maximum speed of the configured system. Replica nodes make other nodes to think that a replica node u moves faster than the predefined speed. In this case, they can detect u as a replica node. Disadvantages of these algorithms are (1) high communication overhead as each node should broadcast its identity and its neighbors' identity periodically; (2) maximum and minimum speed of nodes should be predefined; and (3) each node should be aware of its location.

Dimitriou et al. [22] developed two distributed schemes to defend against replication attack in mobile WSNs. The main idea of these schemes comes from this consideration: each two SNs generate random nonce and send them for each other in their first meeting. The nodes store these random nonce in their memory. Upon the next meeting, they ask each other for the random nonce they exchanged in their previous meeting. If a node cannot reply or replies with a wrong nonce then it is treated as a malicious node and the ID of the node is considered compromised.

Zhou et al. [23] proposed a hybrid local and global detection algorithm to defend against replication attacks in mobile WSNs. The algorithm uses the local detection in a local area smaller than the whole network area to improve the meeting probability of contradictory nodes, while the global detection is used to identify the far away replicated nodes in larger area. This algorithm is based on location claims and selecting some checking locations to verify the location claim messages. This scheme is also described in more details and vast experiments in [24]. Disadvantages of this algorithm are (1) high memory and communication overheads; and (2) the need for determining location of nodes during the network lifetime.

The main idea of the algorithm [25] is to use a pre-distributed pair-wise key protocol based on polynomials and Bloom filters to guarantee that replica nodes cannot be located close to their real identities and collect some pair-wise keys established by each node. The algorithm is centralized and the base station is responsible to identify replica nodes by looking at whether the number of pair-wise keys established by them exceeds a threshold. Disadvantages of this algorithm are (1) being centralized (not being scalable); (2) long detection process for marking replica nodes; and (3) high communication overhead.

Conti et al. [26] proposed History Information-exchange Protocol (HIP) and its optimized version (HOP) to detect replication attacks. Both of these protocols employ local information and node mobility to detect replica nodes. These algorithms run in many rounds. In each round, nodes broadcast their location claims to perform location information exchange. Each node stores the location claims of its neighbors for at least r rounds, in a history log. Neighboring nodes exchange and compare their history logs. The HIP and HOP differ in how they execute the logs comparison. In both algorithms, each node compares its own log with logs received from its neighbors. However, in HOP each node also compares the received logs between them. Therefore, HIP and HOP differ in the amount of computation required. Disadvantages of these algorithms are (1) high memory/computation/communication overheads; and (2) the need for determining the location of nodes during the network lifetime.

Manickavasagam and Padmanabhan presented a distributed algorithm in [27] to detect replica nodes in mobile WSNs. This algorithm is based on the idea of sequence elements, which uses the Sequential Probability Ratio Test (SPRT) to make a decision. The algorithm uses the fact that when several replica nodes transmit their message simultaneously, multiple messages with the same source ID but physically different sources are propagating in the network. The algorithm attaches a sequence appendix to every transmitted message, to provide a sort of message ordering. When an intermediate node encounters an out of order message, it suspects that the source of this message may be a replica node. This algorithm suffers from high memory and communication overheads.

Yu et al. [28] proposed Efficient and Distributed Detection (EDD) algorithm to detect replica nodes in mobile WSNs. The main idea of EDD comes from the fact that in a mobile WSN without replica nodes, the number of times that node u faces a specific node v should be very limited, in a period T of network lifetime. Also, in a network with two replica nodes v, the number of times that node u faces node v in a period T of network lifetime should be larger than a threshold. Disadvantages of this algorithm are its high memory and communication overheads.

Jamshidi et al. [29] proposed a security algorithm based on learning agents and WNs to defend against replica nodes in mobile WSNs. The algorithm employs a few WNs which are equipped with a learning agent. WNs monitor the network traffic and other nodes' mobility and update their learning agent when meeting other nodes. The detection phase of the algorithm runs after r monitoring rounds in which WNs mark replica nodes according to their learning agent's status. The drawbacks of this algorithm are low convergence speed in the learning agents (particularly when there are many nodes in the network), low detection rate, and lack of scalability (by increasing the number of nodes in the network, detection probability decreases significantly).

Jamshidi et al. [30] also proposed another watchdog based algorithm to defend against replication attacks in mobile WSNs. This algorithm uses nodes' speed and the maximum predefined speed for SNs as a factor to make decision about nodes' nature. WNs walk in the network and record time-location tags in their buffer when they meet SNs. WNs send their history tags when they meet and collaborate to measure nodes' speed in the network. If WNs find that a node moves faster than a predefined threshold, they consider it as a replica node, because such replica node in different regions of the network is moving faster than usual in different regions of the network. The drawbacks of this algorithm are low speed (particularly when there are many nodes in the network), independence of WNs, high memory and computation overheads.

The proposed algorithm employs a few WNs in the network to monitor the traffic and mobility of SNs in a silent manner. Each WN keeps a vector of size $O(n)$ so that, each element i of the vector shows the probability of meeting node SN_i. All elements of the vector are initially set to $^1/_n$ and are gradually updated according to the neighboring information. That is, the WN increases the value of the element i if SN_i is in the neighborhood range of the WN; otherwise, the value of the element i is decreased. The proposed algorithm is not centralized and does not need to determine the location of nodes, broadcasting location claim messages, and complicated replica node detection procedures. Although the proposed algorithm slightly increases the memory and computation overhead of the WNs, it does not impose any communication, computation, or memory overhead on resource-constraint SNs. Moreover, the scalability and high detection accuracy of the proposed algorithm make it superior to the existing algorithms.

3. System assumptions and attack model

The network includes a set of SNs (i.e., normal nodes) and a separate set of WNs which are randomly deployed in a 2D environment. The number of WNs is much smaller than the number of SNs (i.e., $|WN| \ll |SN|$). The total number of nodes in the network is $N = |WN|+|SN|$. SNs are responsible for the network mission (collecting information, transmitting data to the base station, etc.) and WNs are responsible for detecting replica nodes. Each SN has a unique ID. They are mobile (following the mobility mobiles such as waypoint model) but they have no idea about their real-time location. All SNs and WNs employ omnidirectional dissemination and have the same transmission range. Unlike WNs which are tamper-resistant, SNs are not protected. That is, if they are captured by an adversary, their secret information might be revealed. Moreover, the adversary may can reprogram them [31,32]. Since SNs are mobile, they should periodically (after each time period t or when they reach to a new location in the network) propagate a "Hello" packets, route request tables, data packets, and keep alive messages. This is one of the requirements of a mobile WSNs through which mobile SNs can detect their neighbors, establish the secrete keys, discover new routes, and communicate with the base station or other SNs [2,3,18]. To avoid interfering with the network operation, WNs remain hidden from SNs. That is, WNs are prevented from sending messages until they detect the replica node(s). The system assumptions are summarized in Table 1.

Table 1
System assumptions.

The network consists of SNs and WNs.
All nodes are deployed in a 2D environment, randomly.
The number of WNs is much smaller than the number of SNs ($\|WN\| \ll \|SN\|$).
The SNs are responsible for the network mission and the WNs are responsible for detecting replica nodes.
Each SN has a unique ID and is not aware of its location.
All SNs and WNs employ omnidirectional dissemination with the same wireless transmission range.
SNs and WNs are mobile and move in the network according to the mobility models such as random waypoint.
Despite WNs which are tamper resistant, SNs can be decoded and reprogrammed.
In order to update the neighborhood and routing tables, SNs should periodically propagate a "Hello" message, route request, data sending or keep alive messages. But WNs are prevented from sending such messages so that they remain hidden from malicious nodes as well as SNs.

Table 2
Attack model.

The network is developed in an adversary environment and thus, it is insecure.
The adversary can capture some of SNs and distribute clones of them, as replica nodes, in the network.
The replica node(s) have the same ID(s) as the captured SN(s).
The adversary cannot generate new valid IDs to be used by the replicas.
Similar to normal SNs, all replica nodes periodically propagate "Hello" packets, routing requests, data packets, or keep alive messages.
The replica nodes can be mobile like normal nodes or the adversary might seat them in specific locations.

It is also assumed that the network is developed in an adversary environment, thus, it is insecure and adversary can capture some of SNs and create copies and distribute them. We also assume that the adversary does not generate new IDs different from the ID(s) of the captured SN(s) [27]; otherwise it would be easy for WNs to detect the malicious nodes. The replica nodes can be mobile like normal nodes or the adversary might seat them in specific locations. Table 2 summarizes the attack model.

4. The proposed algorithm

As mentioned before, each WN iteratively overhears the network traffic for periods of t seconds. Each iteration is called a monitoring round and is represented by r. The main idea of the proposed algorithm is based on a simple fact that for a uniformly distributed set of mobile SNs in the network which are moving according to the mobility models such as waypoint mobility model, all SNs have the same chance to be met by a WN at each monitoring round. Let D be the average number of nodes a WN is met at each round. In other words, D is the average number of neighbors of a WN. D is a function of transmission range of the SNs and the network density, thus, it is constant over time as neither of the SNs' transmission range nor the network density is changed over time. Accordingly, in a network of n mobile SNs with n unique IDs (ranging from 1 to n), the probability that SN_i (which represents the SN with ID $= i$ and $1 \leq i \leq n$) is met by a WN is D/n. Con-

sequently, it is expected that after a several monitoring rounds, a WN has met all SNs almost the same number of times.

When a replica attack happens, the attacker generates λ copies of each captured SN_u (where u is the ID of the captured SN) and distributes them in the network area. Since the malicious nodes use the ID of the captured node and benefits from its keying information and functions, SNs cannot detect them. Motivated by the above discussion, the following theorem is used to detect the replicated ID in a lightweight but efficient way.

Theorem 1. *Consider a WSN with n mobile SNs where each SN_i has a unique ID = i ($1 \le i \le n$). If the attacker captures a node SN_u and distributes λ copies of it in the network, we would have one of the following observations:*

(1) If $\lambda = 1$, the probability that SN_u is observed by a WN is equal with that of all other SNs. In other words, when no attack happens or when the attacker returns the replica node to the network without generating extra copies, the SNs have the same chance to be met by the WNs.
(2) If $\lambda > 1$, the probability that the captured ID (i.e., the nodes with ID = u) is met by WNs is higher than that of the uncaptured IDs.

The proof of Theorem 1 is given in the Appendix. However, the theorem is reasonable as the IDs with higher frequencies (because the replicated nodes have the same ID) have higher chance of being met by an observer compared to uncaptured nodes. Motivated by Theorem 1, a three-phase mechanism is presented what follows to detect the replicated IDs. Fig. 2 depicts the flowchart of the proposed mechanism. In a high level view, the first phase configures the WN(s). In the *second phase*, each WN periodically monitors the network traffic, records the observation history, and gives a rough estimation of the probability of observing each SN. In the *third phase*, WNs detect replica nodes regarding the probabilities estimated in the second phase. The details of these three phases are given in the rest of this section.

4.1. The first phase (configuration of WNs)

Before deployment of SNs in the network, WNs are configured. Each WN has a $n \times 1$ vector called Probability of Observation vector (or *PO* for short) as is shown in Fig. 3 so that, the ith row of the *PO* vector represents the expected probability of observing SN_i and is shown by PO_i. Thus, it is initially set to $1/n$ and $\sum_{i=1}^{n} PO_i = 1$. Besides, a WN uses a vector to keep the list of ID of its real-time neighbors. This vector is called the List of Neighbors and is represented by *LN* for short. WNs can use dynamic memory allocation technique to efficiently manage the memory allocated to their *LN*.

4.2. The second phase (traffic/mobility monitoring)

Monitoring round is performed by WNs during the network lifetime. During each round r, SNs periodically propagate "Hello" messages to introduce themselves to the new neighbors. They may send route request messages, data packets, and even keep alive messages if required. This allows the WNs to silently overhear the channel and gradually update the *LN* vector by filling it out with the ID of the new neighbors. At the end of each round r, each WN updates the *PO* of all SNs in the following way:

(1) It decreases the *PO* of SN_i (i.e., PO_i) if the SN_i is not in the *LN* at the end of round r. To do so, WN uses a very small positive penalty factor α ($0 < \alpha \ll 1$) and updates the PO_i according to the Eq. (1):

$$PO_i = (1-\alpha)PO_i \quad \forall i \notin LN \tag{1}$$

(2) It accumulates the reduced values (i.e., the penalties) in an accumulator (σ) as below:

$$\sigma = \alpha \sum_{\forall i \notin LN} PO_i \tag{2}$$

(3) Then, the accumulated penalty is equally shared between all observed IDs (i.e., the IDs in the *LN*).

$$PO_i = PO_i + \frac{\sigma}{|LN|} \quad \forall i \in LN \tag{3}$$

It is expected that after a while (e.g., t unites of time), the network topology has been changed due to the mobility of the SNs and WNs. Hence, the second phase is iteratively executed. That is, WNs and SNs probably update their location, and WNs update their *LN* and *PO* matrices.

Considering the second phase, for a constant number of replica nodes, it is expected that the *PO* corresponding to the ID of replica nodes exceeds its initial value (i.e., $1/n$) as time passes (i.e., as r is increased), while the *PO* of other SNs decreases. This is reasonable as the ID of the replica nodes has a higher chance to be met by the WNs. To show how the *PO* of the replica node is changed over time, we have run a simulation on a network of 96 SNs and 4 WNs. It is assumed that $\alpha = 0.01$, the average number of neighbors is $D = 12$, and the adversary captures SN_1. The simulation is performed for 1000 rounds. In order to consider the impact of the number of replica nodes on *PO*, the simulation is repeated for a different number of replicas from $\lambda = 1$ to $\lambda = 11$. Fig. 4 shows the average value of *PO* of SN_1 calculated by four WNs over 1000 rounds. The results show that the *PO* of SN_1 remains almost unchanged over time, when there are no replicas (i.e., when $\lambda = 1$). In contrast, there is a significant growth in *PO* as the number of replica nodes is increased.

4.3. The third phase (detection of replica nodes)

After completion of the second phase, WNs run the third phase of the proposed algorithm to detect replica nodes, independently. As explained in Theorem 1 and proved in the Appendix, PO_i ($1 \le i \le n$) is expected to be $1/n$ if there is no replica node; otherwise, PO_i is higher than $1/n$ if SN_i is captured by the adversary, or it is below $1/n$ if SN_i is a normal node. In the other words, if there are several copies of a node, like SN_i, in the network, the number of times that a node with ID i appears in the neighborhood of a WN would be higher than that of other SNs [28]. To show this, we have run another simulation on the described network in Section 4.2. Here, we have considered three different scenarios: (1) there are no captured SN and thus, no replica nodes; (2) the adversary captures only one SN with ID = 1 (i.e., SN_1) and creates 10 copies of it ($\lambda = 10$); and (3) the adversary captures five nodes with ID = 1–5 (i.e., SN_1, SN_2, SN_3, SN_4, SN_5) and creates 10 copies of each of them (totally 50 replica nodes will be distributed in the network). The simulation is performed for 1000 rounds (i.e., from $r = 1$ to $r = 1000$). Fig. 5 shows how the PO_i ($1 \le i \le n$) is changed over time under the three mentioned scenarios. The results show that the PO_i is almost $1/n$, when there is no replica node. In the second and the third scenario, where replica nodes are distributed in the network, the *PO* of the replica nodes exceed $1/n$, while the *PO* of other SNs is reduced to below $1/n$.

Motivated by this, a simple and lightweight procedure is to consider SN_i as a replicated node or consider i as a replicated ID if PO_i is greater than $\Delta \times 1/n$. Parameter Δ should be selected greater than 1 because, for establishing a replication attack, the adversary must inject at least two replica nodes to the network. So, it is expected that the *PO* of the replica node should be greater than $1/|SN|$.

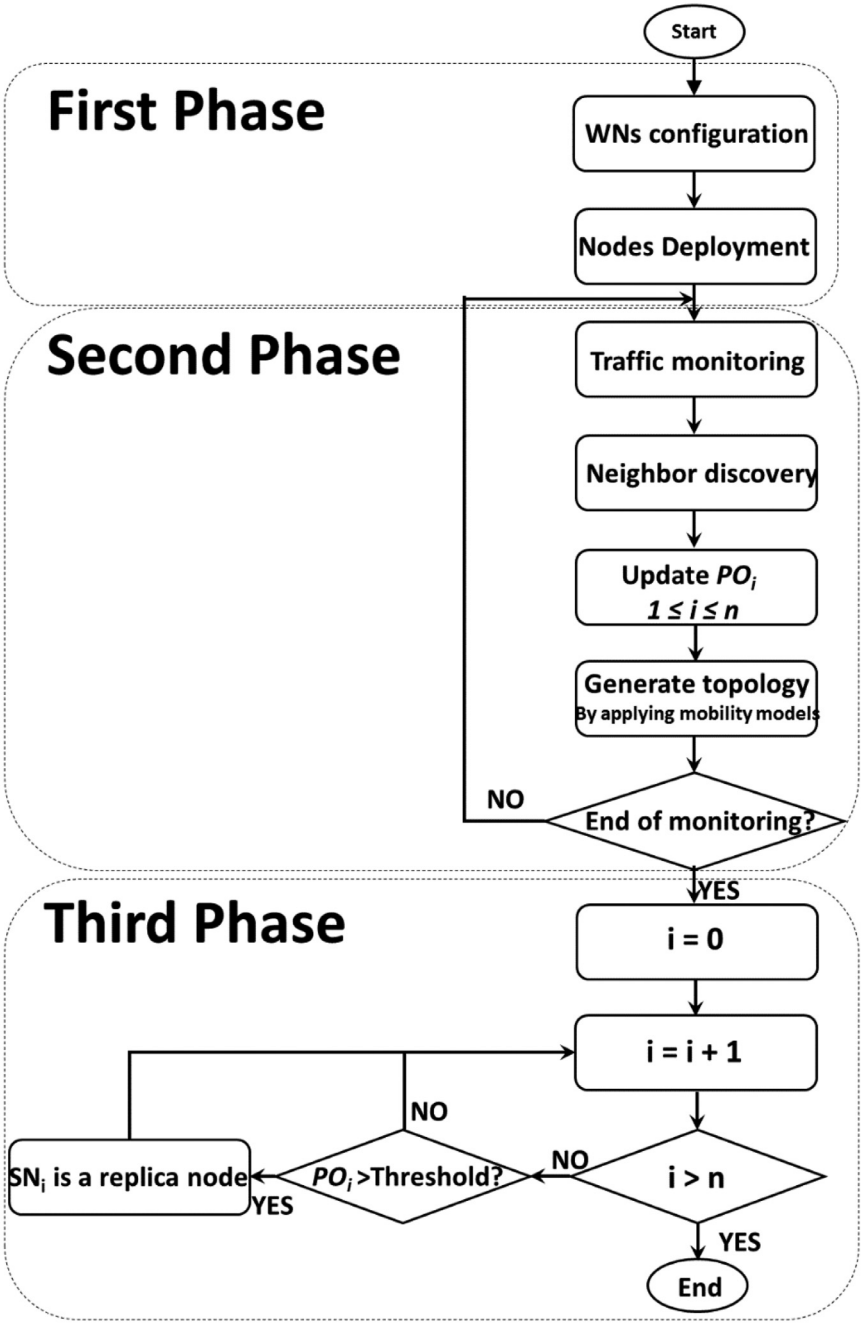


Fig. 2. The flowchart of the proposed algorithm.

In this phase, each WN creates a replica detection list, named RDL_{WN_i}, containing the replica identifiers that it has detected, then sends it to the base station, in a secure manner [33]. The base station simply marks all identifiers of all received RDL_{WN_i}as replica IDs. That is, an ID would be marked as a replica ID if at least one WN detects it as a replica node. The base station may propagate an alarm message to the whole network in order to prevent normal nodes from communicating with nodes that has these replica IDs.

5. Efficiency analysis and simulation results

In this section, the efficiency of the proposed algorithm in terms of communication, memory, and computation overhead is investigated and the results are compared with the state-of-the-art algorithms. Then, the simulation results of the proposed algorithm are presented.

5.1. Efficiency analysis

Memory Overhead: Since the proposed algorithm is only run by WNs, no memory overhead is imposed on normal nodes. But WNs require some memory blocks on the order of $O(n)$ to store the PO, and another memory blocks on the order of $O(D)$ to store LN vector, which is negligible in comparison to the $O(n)$. Therefore, the memory overhead of each WN is of order $O(n)$. Please note the proposed mechanism does not impose extra memory overhead to the SNs. Hence, the memory overhead of the proposed algorithm is $O(|WN| \times n)$, per network. Table 3 compares the memory overhead of the proposed algorithm with other existing algorithms, per network. As Table 3 shows, the proposed algorithm and [29] are

Table 3
Performance comparison of the proposed algorithm and the other algorithms in terms of memory, communication, and computation overhead [5,9,12,18,28,37].

Algorithm	Network type	Algorithm type	Technique	Need additional nodes	Memory overhead (per network)	Communication overhead (per network)	Computation overhead (per node)
LSM [7]	Stationary	Distributed	Location claim based	No	$O(n\sqrt{n})$	$O(n\sqrt{n})$	$O(\sqrt{n})$
RED [8]	Stationary	Distributed	Location claim based	A satellite or UAV	$O(nD)$	$O(n\sqrt{n})$	$O(\sqrt{n})$
RAWL [9]	Stationary	Distributed	Location claim based	No	$O(n\log n\sqrt{n})$	$O(n\log n\sqrt{n})$	–
ID-based [10]	Stationary	Distributed	Multiple-sink based and dynamic ID assignment	Several sink nodes	$O(n)$	$O(n)$	$O(1)$
CSI-1 [11]	Stationary	Centralized	Compressed Sensing-based	No	$O(n\log n)$	$O(n\log n)$	–
CSI-2 [11]	Stationary	Centralized	Compressed Sensing-based	No	$O(n)$	$O(n)$	–
Scheme II [13]	Stationary	Distributed	group deployment and key based	No	$O(nD)$	$O(n\sqrt{n})$	$O(n)$
Scheme III [13]	Stationary	Distributed	Group deployment and key based	No	$O(nD)$	$O(n\sqrt{n}\log m)$	$O(n)$
Algorithm [14]	Stationary	Distributed	Group deployment and RSSI based	No	$O(n)$	$O(nD+n\sqrt{n})$	$O(Dr)$
XED [18]	Mobile	Distributed	Random number and node meeting based	No	$O(n^2)$	$O(nDr)$	$O(Dr)$
SPRT [20]	Mobile	Centralized	Speed and node meeting based	No	$O(n^2)$	$O(n\sqrt{n})$	$O(nrbp)$
TD [24]	Mobile	Distributed	Location-based and node meeting based	No	$O(nD^2r)$	$O(nD^2r)$	$O(D^3r)$
Algorithm [25]	Mobile	Centralized	Key based	No	–	$O(n\log n)$	$O(nr)$
HOP [26]	Mobile	Distributed	Location-based and node meeting based	No	$O(nD^2r+nDr)$	$O(nD^2r)$	$O(D^3r)$
Optimized SPRT [27]	Mobile	Distributed	Sequence appendix is attached to every transmitted message	No	$O(n^2)$	$O(nr)$	$O(r\sqrt{n})$
EDD [28]	Mobile	Distributed	Speed and node meeting based	No	$O(n^2)$	$O(nr)$	$O(r\sqrt{n})$
Algorithm [29]	Mobile	Partially Distributed	Using WNs and learning agents	Several watchdogs	$O(\|WN\|n)$	0	$O(nr+n\log n)$
Algorithm [30]		Partially Distributed	Using WNs and time-location tags	Several watchdogs	$O(\|WN\|n)$	$O(r\|WN\|)$	$O(n^2r)$
Proposed algorithm	Mobile	Partially Distributed	Using WNs and node meeting based	Several watchdogs	$O(\|WN\|n)$	0	$O(nDr)$

n: the number of SNs in the network.
D: the average number of neighbors of a node.
bp: the number of location claims which are sent by every node to the base station.
r: monitoring rounds.
$|WN|$: the number of WNs in the network.
m: the number of sensor groups.
Hint: $|WN| \ll n$ and $|WN| \ll D$

	PO
1	$1/n$
2	$1/n$
.	.
.	.
.	.
n	$1/n$

Fig. 3. *PO* vector of WNs; each element is initially set to $1/n$.

more memory efficient compared to the other algorithms for mobile WSNs. It worth considering that the number of WNs is less than the average number of neighbors of a node (i.e., $|WN| < D$), and much less than the total number of nodes in the network (i.e., $|WN| < n$). Of course, the memory overhead of algorithms [10,12,16] is on the order of $O(n)$, but these algorithms do not support mobile WSNs.

Computation Overhead: since the proposed algorithm is executed by WNs, no computation overhead is imposed on SNs. But, each WN should update its *PO* vector in $O(n \times D)$ after each round of the monitoring phase. In addition, each WN in the third phase requires $O(n)$ to scan its *PO* vector to detect replica nodes. Thus, computation overhead of the WNs is of order $O(r \times n \times D)$. Table 3 also compares the computation overhead of the proposed algorithm and the other algorithms.

Communication overhead: Considering energy constraint of SNs, the energy consumption of the proposed algorithms is very crucial for WSNs. It is worth noting that sending, receiving, and processing a packet are the main energy-consuming tasks in a SN. Hence, the total number of transmitted packets under the use of an algorithm provides a meaningful measure of the energy-consumption of the algorithm [3,13,14,25,29,33–37]. Since the proposed algorithm only employs *Hello, route request, data sending*, and *keep alive* messages and since transmitting these messages is one of the requirements of the mobile WSNs (apart from presence or absence of the proposed algorithm), no communication overhead is imposed on SNs. Moreover, WNs send no messages during the execution of the proposed algorithm, and thus, no communication overhead is also imposed on these nodes. Table 3 shows that the communication overhead of other algorithms is very high. Since the proposed algorithm does not impose any communication overhead on WNs, it is also an energy-efficient algorithm.

5.2. Simulation results

The proposed algorithm is implemented using J-SIM simulator [38] and its efficiency is evaluated in terms of true detection probability and false detection probability. Finally, the efficiency of the proposed algorithm is compared with the algorithms presented in [8,9], and [25].

- Detection Probability (P_d) is the probability of detecting replica nodes in the network.
- False detection probability (P_f) is the probability of detecting a non-replica node as a replica node.

In the simulation setup, it is assumed that the network contains n SNs which are distributed in a 100 × 100 m^2 area. The adversary captures M normal nodes of the network and generates λ replicas from each of them. Therefore, the network contains $M \times \lambda$ malicious nodes. In addition, the number of WNs is $|WN|$. The radio range of all nodes is set to 15 m (except for scenarios 7 and 8). The number of rounds in the second phase of the proposed algorithm is r. We used the two-dimensional IID mobility model [3,20] to verify the results. In the IID mobility model, at beginning of each round, sensors are uniformly and randomly distributed in the network, so that the current position of each node is indepen-

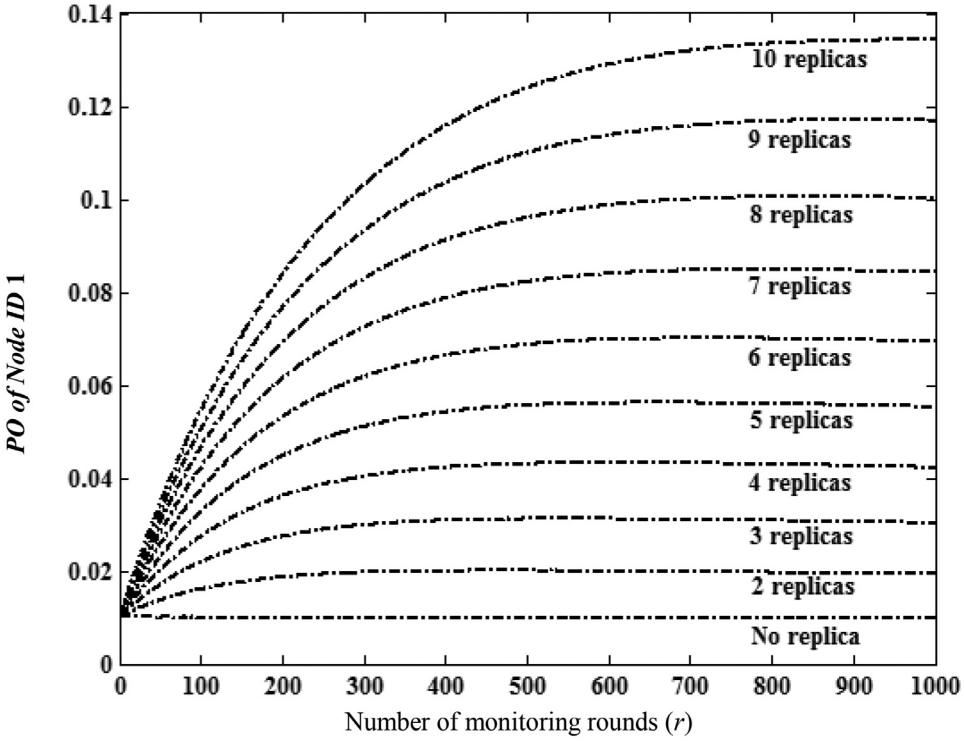


Fig. 4. *PO* of replica node in different scenarios with $|WN| = 4$, $n = 96$, $D = 12$, and $\alpha = 0.01$.

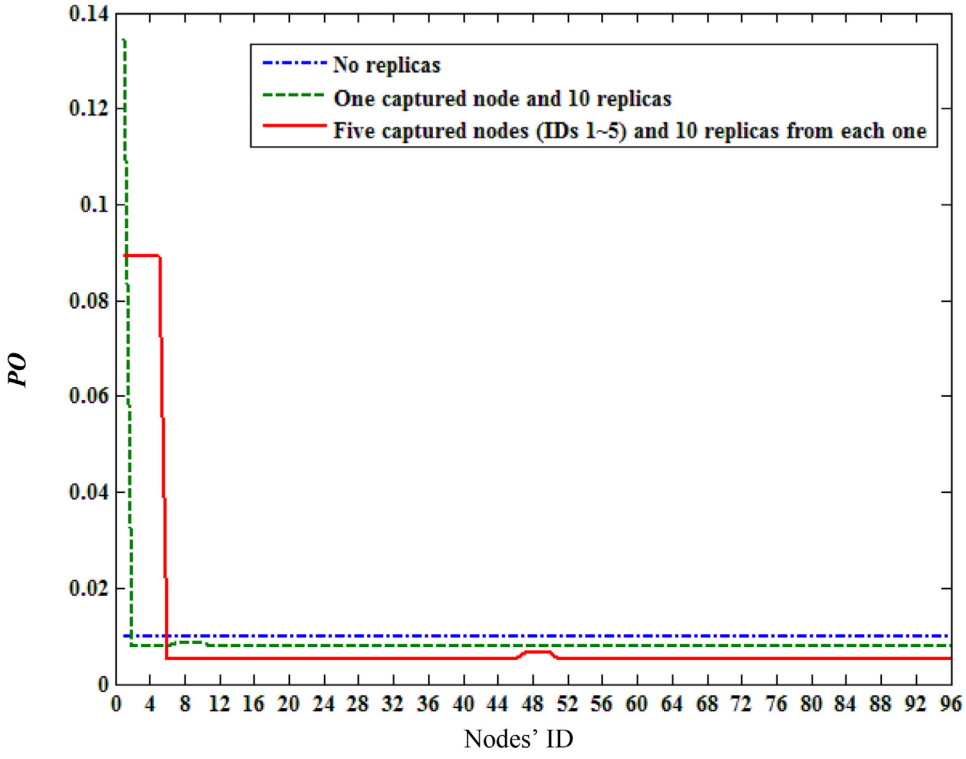


Fig. 5. PO_i $(1 \leq i \leq n)$ under three scenarios (1) no captured an replicas nodes, (2) one captured node (ID = 1) with 10 copies of it, and (3) 5 captured nodes (IDs = 1~5) with 10 copies of each one ($r = 1000$, $|WN|=4$, $n = 96$, $D = 12$, $\alpha = 0.01$).

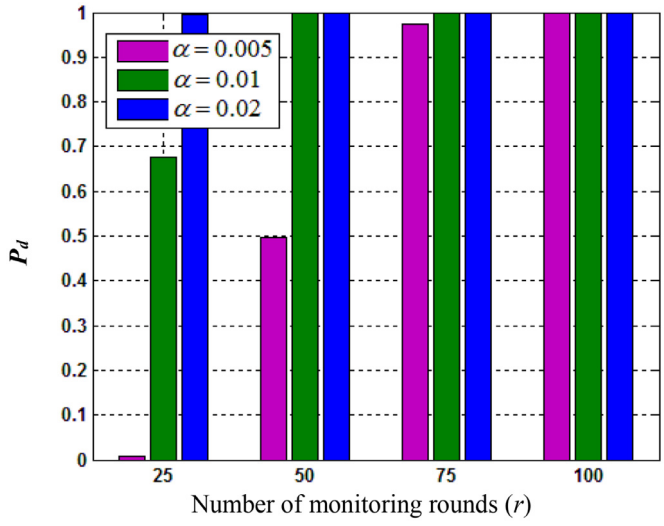


Fig. 6. The impact of$\alpha$and r on P_d of the proposed algorithm.

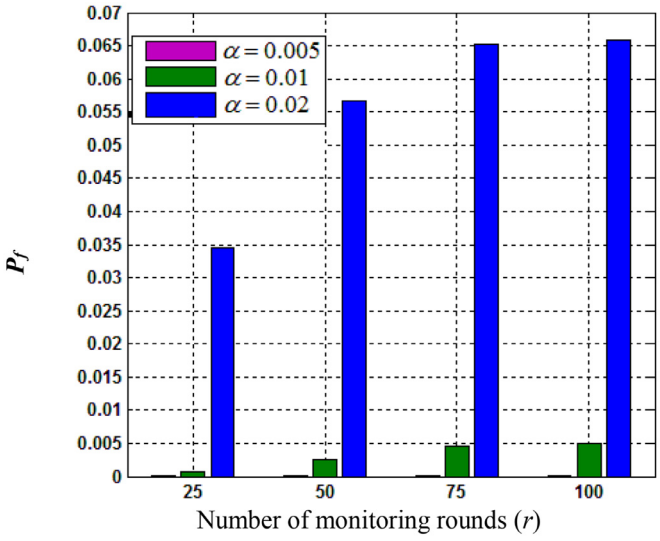


Fig. 7. The impact of$\alpha$and r on the P_f of the proposed algorithm.

dent from its position at the previous round [18,28]. Each scenario is carried out for 100 times, and the results are averaged over 100 trials.

Scenario 1: This scenario investigates the effect of α (reward/penalty coefficient) and r (the number of monitoring rounds) on the detection accuracy of the proposed algorithm. In this scenario, parameters are adjusted as $|WN| = 4, M = 1, \lambda = 5, n = 300, \Delta = 2$. Figs. 6 and 7 respectively show P_d and P_f for the case $\alpha \in \{0.005, 0.01, 0.02\}$ and $25 \leq r \leq 100$. The results show P_d is increased as r is increased. As mentioned earlier, this is reasonable because by increasing r, WNs have higher chance to meet all nodes and thus, the PO_i of the replicas will be increased and the PO_i of other SNs will be decreased. Therefore, in the detection phase of the proposed algorithm, the condition $PO_i > \Delta/\eta$ for replica nodes is satisfied and they will be detected with an ultra-high probability.

Besides, the simulation results show that increasing α to some extent increases P_d. For instance, if $\alpha = 0.005$, P_d for $r = 25$, 50, 75, and 100 rounds will be 0.008, 0.05, 0.97, and 1, respectively. But, if $\alpha = 0.01$, the P_d for $r = 25$ is 0.68 and for $r \geq 50$ is 1. This is because, replica nodes have a higher chance to be met by a WN and thus, their corresponding element in PO vector will be updated more frequently, compared to that of other SNs. Thus, choosing a larger α will speed up the growth of the corresponding element in

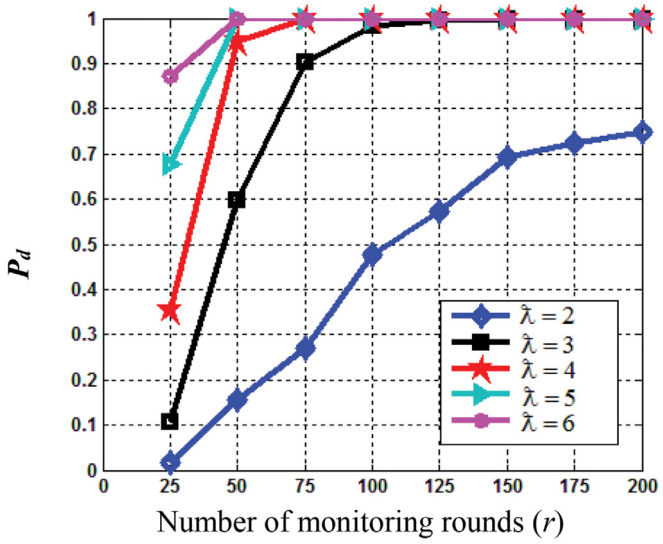


Fig. 8. The impact of λ on P_d of the proposed algorithm.

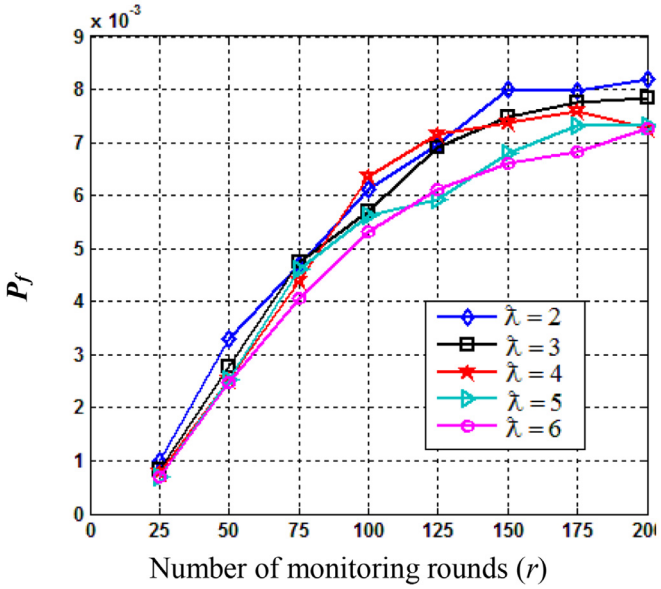


Fig. 9. The impact of λ on P_f of the proposed algorithm.

PO vector. For instance, by setting α to 0.02, P_d reaches to 1 only after $r = 25$ rounds.

On the other hand, as can be seen in Fig. 7, increasing *r*, slightly increases P_f. Because by increasing *r*, it is possible that some non-replica nodes (i.e., normal nodes) might appear many times in the neighborhood of the WNs as "unlucky SNs", which is caused an increase in the corresponding element in the *PO* vector. Thus, in the detection phase, these unlucky SNs are considered as replica nodes incorrectly, because their *PO* is greater than Δ/η.

Increasingαmay negatively affect P_f. That is, a bigαmay sharply increase the corresponding *PO* of a normal SN, when it is accidently met by a WN in several consecutive rounds. For instance, if $\alpha = 0.005$ the P_f for $r < 100$ is below 0.0001 but for $\alpha = 0.02$ it is about 0.066 ($P_f \approx 0.066$). In general, the simulation results show that considering $\alpha = 0.01$ and $r = 50$ is a suitable configuration for the discussed scenario. Because, on one hand, it detects all replica nodes in the network (i.e.,$P_d \approx 1$) and on the other hand, it results in a small false detection probability ($P_f \leq 0.005$).

Scenario 2: The purpose of this scenario is to evaluate the impact of λ on the efficiency of the proposed algorithm. In this scenario, 300 SNs are distributed in the network ($n = 300$) and |*WN*|, *M*, α, and Δ are set to 4, 1, 0.01, and 2, accordingly. Figs. 8 and 9 show the impact of changing λ from 2 to 6 on P_d and P_f, respectively.

An adversary can establish replication attacks and disrupt network function under two cases. In the first case, the adversary captures many of normal nodes of the network and generates only a few copies of each of them (for example, 2, 3 or 4 copies) and distributes them throughout the network. Detecting replica nodes under such conditions is very difficult for many of security algorithms, if not impossible. Similarly, it is both difficult and time consuming for the adversary to capture, decode, reprogram, and control many normal nodes.

In the second case, which is more applicable for the adversary, the adversary captures few SNs but generates many copies (for example, 10 copies) of each of the captured SNs. However, detecting replica nodes under this case is much simpler for most of security algorithms.

In fact, as the number of copies of the captured nodes is increased, their chance of being observed by WNs is increased as well. Thus, the *PO* corresponding to a malicious node with ID = *i* increases faster. Therefore, P_d of the proposed algorithm increases. This is presented in Fig. 8. When there are only two copies of a captured node *u* in the network ($\lambda = 2$), after 25 monitoring rounds P_d is reached to about 0.05 while in the case of $\lambda = 4$ and $\lambda = 6$, P_d is reached to 0.35 and 0.87, respectively. It is worth noting that for $\lambda \geq 5$ and $r = 50$, P_d is reached to 1. In addition, for $r \geq 100$ P_d (for $\lambda \geq 3$) is almost 1.

Moreover, as can be seen in Fig. 9, by increasing parameter λ, P_f is also slightly decreased. For instance, after 100 monitoring rounds, for $\lambda = 2, 3, 4, 5$, and 6, P_f of the proposed algorithm is 0.0061, 0.0056, 0.0063, 0.0056, and 0.0053, respectively. In addition, after 150 rounds of the monitoring phase, if there are only two replica nodes in the network, P_f reaches to 0.008; if there are six replica nodes in the network, P_f reaches to 0.0068. As we mentioned, if there is a large number of copies of a captured node *i* in the network, the node SN_i would appear in the neighborhood of the WNs more frequently compared to normal nodes which results in a larger PO_i compared to other SNs. Therefore, P_f is decreased.

Scenario 3: this scenario investigates the impact of *M* (the number of captured nodes) on P_d and P_f of the proposed algorithm. In this scenario, 300 SNs are distributed and |*WN*|, λ, α and Δ are set to 4, 5, 0.01, and 2, respectively. The simulations are carried out for $M = 1$ to 5. Figs. 10 and 11 show the simulation results. As is shown, for $r \geq 50$, varying the number of captured nodes, *M*, does not affect P_d of the proposed algorithm, which is almost $P_d = 1$. But, for smaller *r*, for instance, $r = 25$, increasing *M*, decreases P_d. Moreover, the simulation results illustrated in Fig. 10. show that increasing *M* results in the increase of P_f as well. This is reasonable because when the adversary captures a large number of SNs and generates λ copies of each of them, nodes with replicated ID appear more frequently in the neighborhood of WNs. Thus, the corresponding element of *PO* vector growths and exceeds the threshold Δ/η thus, P_f is decreased.

Scenario 4: This scenario investigates the effect of Δ (reward/penalty coefficient) on P_d and P_f. In this simulation 300 SNs are distributed in the network and |*WN*|, *M*, λ, α and Δ are set to 4, 3, 3, 0.01, 2~4.

Figs. 12 and 13 show the simulation results in terms of P_d and P_f, respectively.

The results show that decreasing Δ, increases P_d of the proposed algorithm which is reasonable as taking a smaller Δ results in a smaller threshold Δ/η. For example, considering $\Delta = 2$, if the adversary generates 5 copies of each captured node ($\lambda = 5$) and

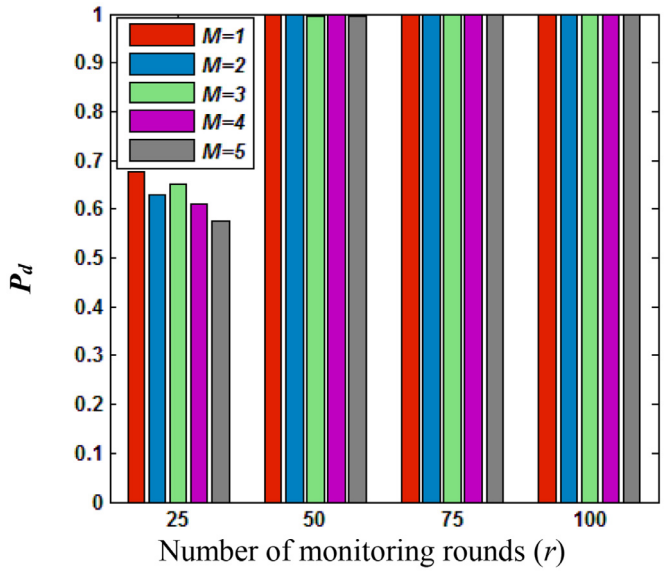


Fig. 10. The impact of M on P_d of the proposed algorithm.

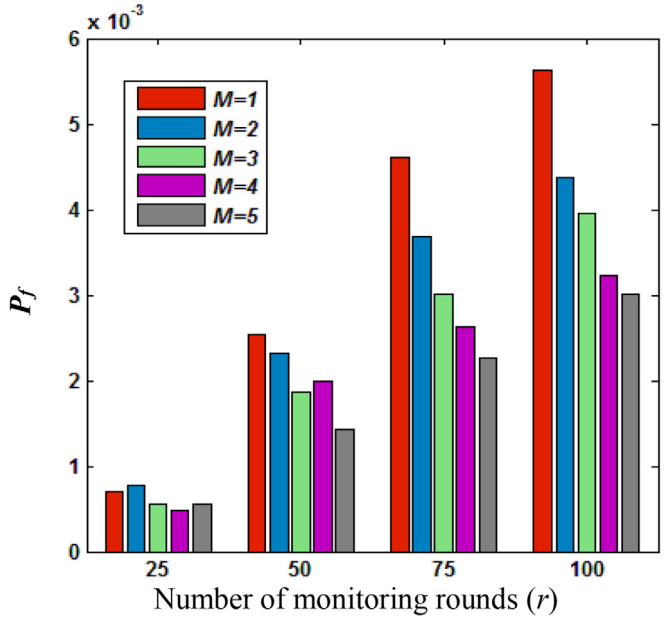


Fig. 11. The impact of M on P_f of the proposed algorithm.

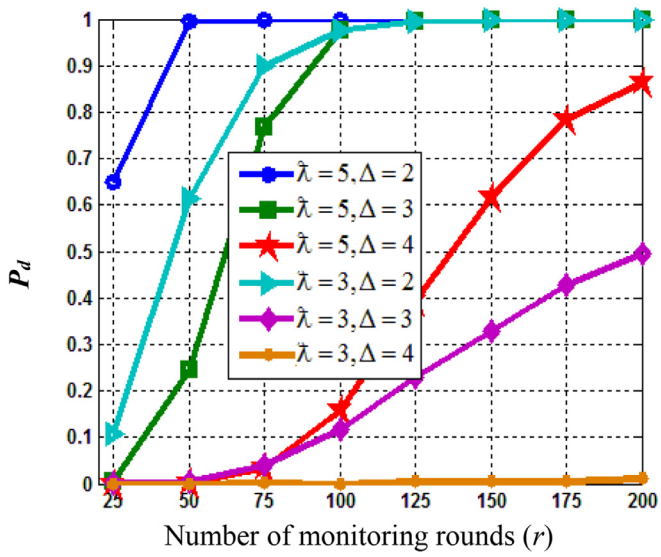


Fig. 12. The impact of Δ on P_d of the proposed algorithm.

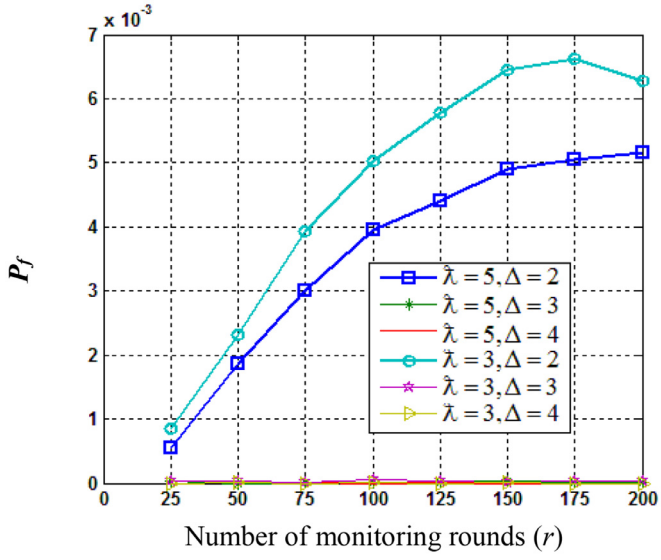


Fig. 13. The impact of Δ on P_f of the proposed algorithm.

distributes them in the network, P_d is reached to almost 1 when r exceeds 50. Under the same setup, P_d is reached to 0.25 and 0.003, when Δ is set to 3 and 4, respectively. But, if $\lambda = 3$, P_d for $\Delta = 2$ would be 0.61; and for $\Delta = 3$ and $\Delta = 4$, P_d would be 0.004, and 0, respectively. In Scenario 2, it was shown that by decreasing λ, P_d is also reduced.

In addition, the simulation results illustrated in Fig. 13 show that by increasing Δ, P_f is also reduced. Because for a normal node SN_i, PO_i is almost $1/n$, the condition $PO > \Delta/n$ would not be satisfied. Even if some normal nodes frequently appear in the neighborhood of WNs, the corresponding PO would not be greater than Δ/n (for a large Δ). For instance, the result of this simulation scenario shows that for $\Delta > 2$, P_f of the proposed algorithm is almost 0.

Scenario 5: This scenario investigates the effect of parameter $|WN|$ (the number of WNs) on detection accuracy of the proposed algorithm. In the simulation setup, parameters are adjusted as $\Delta = 2$, $M = 3$, $\lambda = 5$, $\alpha = 0.01$, $n = 300$ and the performance of the proposed algorithm is evaluated for $|WN| = 1 \sim 4$. Fig. 14 shows the simulation results in terms of P_d and Fig. 15 shows the results in terms of P_f. It is clear that by increasing the number of WNs in the network, P_d of the proposed algorithm increases. Because, more areas of the network are observed by the WNs, simultaneously. Thus, replica nodes are detected faster. On the other hand, by increasing the number of WNs in the network, P_f is also increased. As mentioned, it is probable that some normal nodes (unlucky nodes) appear randomly in the neighborhood of a WN many times, as a result of which, WNs detect such normal nodes as replica nodes incorrectly. Now, if the number of WNs in the network increases, the probability of occurrence of such case also increases. For instance, as can be seen in Figs. 14 and 15, if there is only one WN in the network ($|WN| = 1$), after $r = 25$ rounds of the monitoring phase, P_d and P_f would be 0.21 and 0.0001, respectively. However, if there are $|WN| = 4$WNs in the network, P_d and P_f would be 0.6 and 0.0007, respectively. Indeed, as the number of monitoring rounds increases, for example $r \geq 100$, even if there is

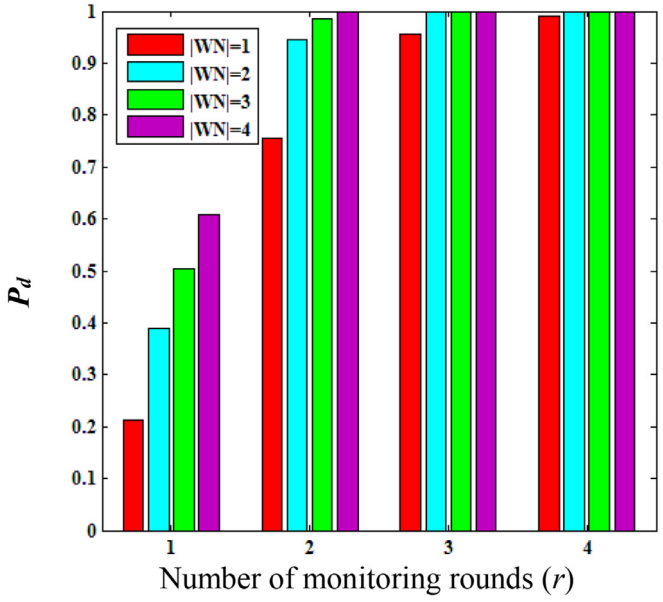


Fig. 14. The impact of $|WN|$on P_d of the proposed algorithm.

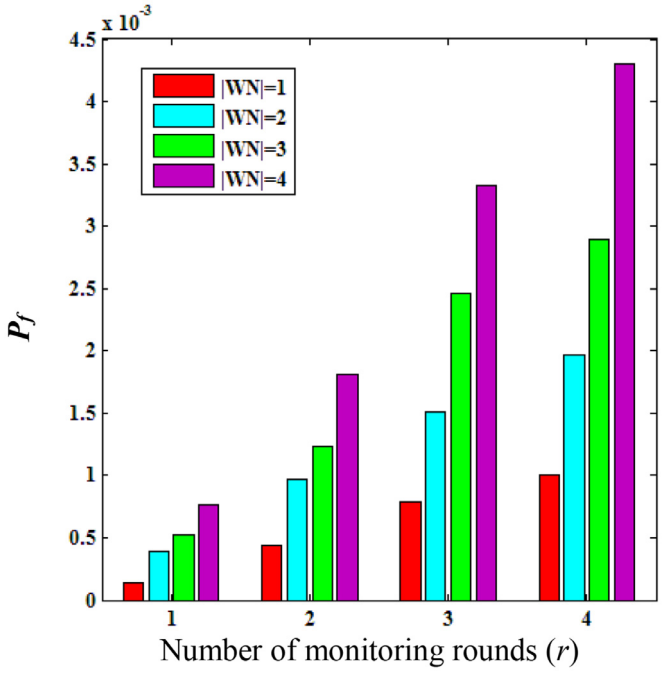


Fig. 15. The impact of $|WN|$on P_f of the proposed algorithm.

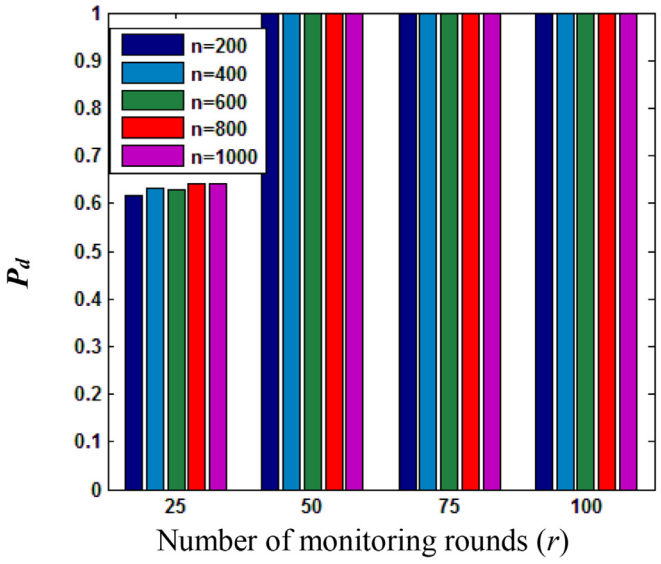


Fig. 16. The impact of n on P_d of the proposed algorithm.

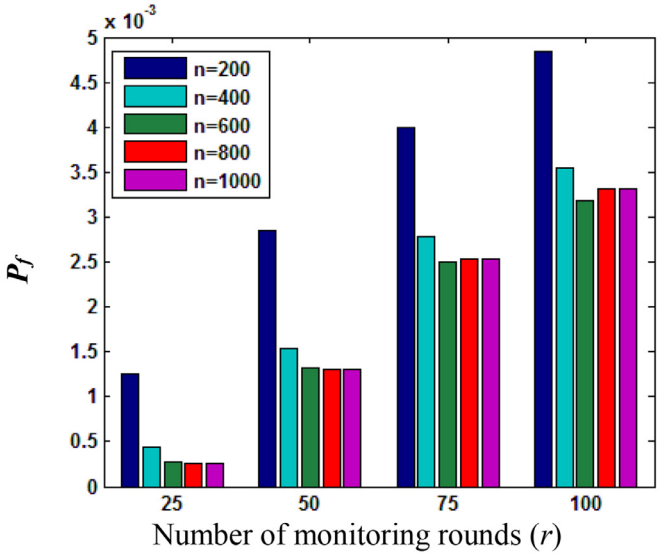


Fig. 17. The impact of n on P_f of the proposed algorithm.

only one WN in the network, all of the replica nodes detected by the proposed algorithm ($P_d = 1$).

Scenario 6: The purpose of this simulation scenario is to investigate the effect of parameter n (total number of nodes in the network) on the efficiency of the proposed algorithm. In this scenario, parameters are adjusted as Δ, M, λ, α, and $|WN|$, to 2, 3, 5, 0.01, and 4. P_d and P_f of the proposed algorithm are evaluated for n = 200–1000. Figs. 16 and 17 show the simulation results in terms of P_d and P_f.

Fig. 16 shows that increasing n does not affect P_d of the proposed algorithm, significantly. The results of this simulation scenario prove the scalability of the proposed algorithm. In addition, Fig. 17 shows that P_f remains almost constant when n exceeds 400. But, when network density is low, P_f increases a little. For instance, if there are $n = 200$ SNs in the network, after $r = 100$ monitoring rounds, P_f is 0.0048 While for $n \geq 400$ P_f is 0.0034. Indeed, it is clear that this difference is negligible; because, if network density is low, WNs might see a few numbers of nodes in their neighborhood at each monitoring round. As a result, at each round of updating *history* matrix of WNs, the *PO* of nodes which have been neighbor of the WN in that round, increases significantly. Therefore, if a SN_i appears in the neighborhood of a specific WN, the corresponding PO_i would be higher than Δ/n and results in the false detection.

Scenario 7: The purpose of this scenario is to compare the efficiency of the proposed algorithm with two of most known algorithms, LSM and RED, which are specific to stationary WSNs. In this simulation scenario, the total number of SNs is considered to be $n = 1000$. Moreover, the most difficult case of establishing a replica node attack is $\lambda = 2$ and $M = 1$. Radio range of nodes is also adjusted such that each node has almost $D = 30$ neighbors. Table 4 shows the adjusted parameters along with the results in terms of P_d and P_f. As can be seen from the results of the simula-

Table 4
Comparing the efficiency of the proposed algorithm to LSM and RED.

Algorithm	parameters	P_d	P_f
LSM	$g = 1, p = 0.1$	0.338	0
RED	$g = 1, p = 0.1$	0.87	0
Proposed Algorithm	$r = 300, \Delta = 2\ \alpha = 0.005, \|WN\| = 10$	0.95	0.005

Algorithm 1 Updating *PO* vector of WNs.

1: $D = |LN|$, $\sigma = 0.0$
2: *for* $i = 1$ *to* n
  if($i \notin LN$)
    $\sigma = \sigma + (\alpha \times PO_i)$
    $PO_i = (1 - \alpha) \times PO_i$
  end if
 end for
3: *for* $i = 1$ *to* n
  if($PO_i \in LN$)
    $PO_i = PO_i + \frac{\sigma}{D}$
  end if
 end for

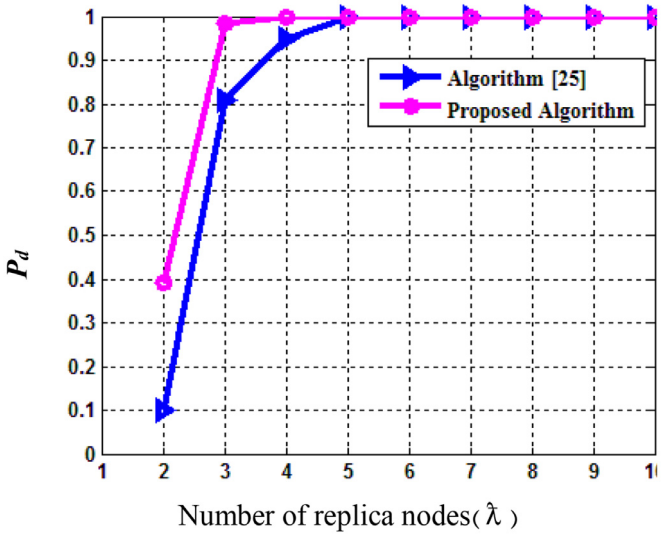


Fig. 18. Comparing the efficiency of the proposed algorithm and algorithm [25] in terms of P_d.

tion, P_d of LSM and RED algorithms are much lower than that of the proposed algorithm. Indeed, P_f of LSM and RED is usually 0% because they use location claims and GPS which are so expensive and energy consuming, while for mobile WSNs (depending on the nature of the algorithm), it is usually greater than 0%. However, P_f of the proposed algorithm is less than 0.005.

Scenario 8: The purpose of this scenario is to compare the efficiency of the proposed algorithm with the algorithm proposed in [25] (specific to mobile WSNs) in terms of P_d and P_f. In the simulation setup, the total number of nodes is $n = 250$ and the radio range of all nodes is selected such that each node has almost $D = 12$ neighbors. In this simulation scenario, it is assumed that the adversary has captured $M = 1$ normal node and has generated and injected $\lambda = 2$~10 copies of the node. In this simulation scenario, algorithm [25] is adjusted with parameters $V_{\min} = 1, V_{\max} = 3, t_{pause} = 20, thershold = 4 \times T(N)$and the proposed algorithm is adjusted with parameters $|WN| = 4$, $\Delta = 2$, $\alpha = 0.01$, $r = 100$. Fig. 18 shows the simulation results in terms of P_d and Fig. 19 shows the simulation results in terms of P_f. The simulation

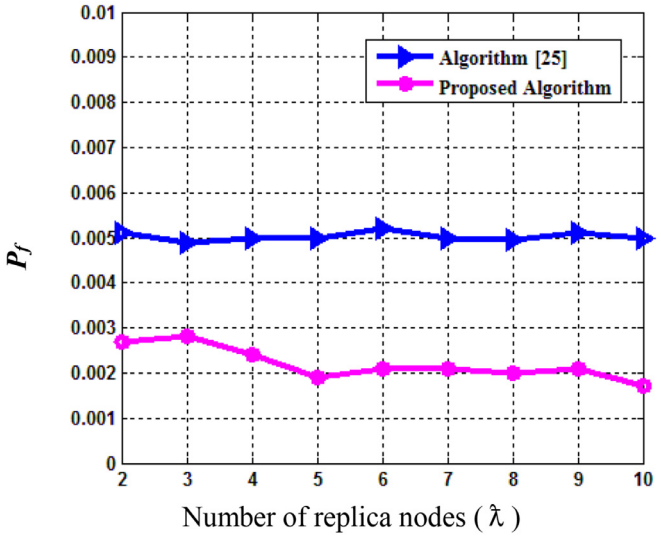


Fig. 19. Comparing the efficiency of the proposed algorithm and algorithm [25] in terms of P_f.

results show that the proposed algorithm outperforms the algorithm [25] in terms of both P_d and P_f.

5.3. Discussion

The proposed algorithm compared to the other existing algorithms has some advantages including:

- It does not impose any overhead on SNs. It only imposes memory and computation overhead on several WNs, but it has no communication overhead.
- It is not centralized and is therefore scalable.
- It is partially distributed and does not suffer from the single point failure.
- It Achieves 100% detection probability while its false detection is about 0.5%.

But the proposed algorithm has some limitations as given bellow:

- It needs additional nodes, called WNs.
- It has a delay from the beginning of the network operation up to the time when it could achieve 100% detection probability. This may cause a slight delay in identifying replica nodes and as a result, giving malicious nodes a chance to perform operations in the network.

6. Conclusion

In this paper, we introduced an efficient though, lightweight replica node detection mechanism that enables the WSNs to detect almost all replica nodes with extremely low false detection probability of 0.005%. The mechanism is designed based on a simple probability rule which is explained as follows: if there is no replica node in the network, all SNs which are identified by their unique IDs have the same chance to be met by an observer. On

the other side, if there are several replica nodes in the network, the replicated ID (i.e., the ID of the captured node which is used by all replica nodes) has a higher chance to be met by an observer. Motivated by this, the mechanism suggests manipulating WNs to silently overhear the network traffic through which, they can find the nodes with abnormally higher chance of being met. To this end, a simple reward/penalty mechanism is used to roughly give an estimation of the possibility of being observed by the WNs. After a while, each WN considers the possibility of meeting a SN and decides whether to advertise its ID as a replica node or not according to a predefined threshold. Through simulating several scenarios using J-SIM we have shown that the proposed mechanism outperforms the state-of-the-art protocols while guarantees no extra computation and memory overhead. Although the proposed mechanism benefits from WNs, it is completely transparent. That is, it neither imposes extra message passing nor interferes with the normal operation of the network. Without interfering with the normal operation of the network and adding extra message passing to the SNs, the proposed mechanism is a practical solution for dealing with a replica attack.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

As mentioned earlier in Theorem 1, we believe that a replica node i has a higher chance to be met by *WNs*. Thus, under the use of the proposed mechanism, PO_i is expected to be higher than that of normal nodes. This section mathematically proves the correctness of Theorem 1.

For the sake of simplicity, we assume there is only one *WN* in the network and only one ID has been replicated. Let i and λ be the ID and the number of replica nodes, respectively, and j $(1 \le j \le n, j \ne i)$ be the ID of a normal node. Considering D as the average number of neighbor nodes around the WN, P_i and P_j are the probability that the *WN* meets a node with ID $= i$ and ID $= j$, respectively.

$$P_i = 1 - \frac{\binom{n-1}{D}}{\binom{n+\lambda-1}{D}} \tag{4}$$

$$P_j = \frac{D}{n+\lambda-1} \tag{5}$$

During each round the *WN* overhears the channel to detect the real-time neighbors, it updates the *PO* vector. That is, it increases the *PO* values corresponding to the observed IDs and decreases the *PO* values corresponding to the other IDs. The updating process is done on a recursive basis as presented in Eqs. (1) and (3). Accordingly, the average value of PO_i at the end of round r (which is shown by $PO_i(r)$) is presented below:

$$PO_i(r) = PO_i(r-1) + \delta_i(r-1) \tag{6}$$

where $\delta_i(r-1)$ is the average reward/penalty at round r added to the $PO_i(r-1)$ and is calculated as follows:

$$\delta_i(r-1) = P_i \times \frac{\sigma(r-1)}{D} - (1 - P_i) \times \alpha \times PO_i(r-1) \tag{7}$$

where $\sigma(r-1)$ is the average amount of penalty, reduced from the nodes at round r - 1.

$$\sigma(r) = \alpha \left(\left(\sum_{\substack{j=1 \\ j \ne i}}^{n} \left(1 - P_j\right) PO_j(r) \right) + (1 - P_i) PO_i(r) \right) \tag{8}$$

By replacing P_i with Eq. (4) and substituting Eq. (8) for $\sigma(r)$, Eq. (7) can be rewritten as follows:

$$\begin{aligned} \delta_i(r) = \frac{\alpha(n+\lambda-D-1)}{(n+\lambda-1)} \Bigg(& \frac{1 - PO_i(r)}{D} + \frac{\binom{n-1}{D}}{D\binom{n+\lambda-2}{D}} PO_i(r) \\ & - \frac{\binom{n-1}{D}}{\binom{n+\lambda-2}{D}} \times \frac{n+\lambda-D-1}{D(n+\lambda-1)} (1 - PO_i(r)) \\ & - \frac{\binom{n-1}{D}^2}{D\binom{n+\lambda-1}{D}\binom{n+\lambda-2}{D}} PO_i(r) - \frac{\binom{n-1}{D}}{\binom{n+\lambda-2}{D}} PO_i(r) \Bigg) \end{aligned} \tag{9}$$

In the same way, the *WN* updates $PO_j(r)$. According to Eqs. (1) and (3), the average value of $PO_j(r)$ is:

$$PO_j(r) = PO_j(r-1) + \delta_j(r-1) \tag{10}$$

where $\delta_j(r)$ is the average reward/penalty at round r added to the $PO_j(r-1)$ and is calculated as follows:

$$\delta_j(r) = P_j \times \frac{\sigma(r)}{D} - \left(1 - P_j\right) \times \alpha \times PO_j(r) \tag{11}$$

By replacing P_j with Eq. (5) and substituting Eq. (8) for $\sigma(r)$, Eq. (11) can be rewritten as follows:

$$\delta_j(r) = \frac{\alpha(n+\lambda-D-1)}{(n+\lambda-1)^2} \left(\frac{\binom{n-1}{D}}{\binom{n+\lambda-2}{D}} PO_i(r) - \lambda \times PO_j(r) \right) \tag{12}$$

It is worth noting that the total reward/penalty added or reduced from the elements of the *PO* vector should remain zero to satisfy the assumption $\sum_{j=1}^{n} PO_j(r) = 1 \ \ \forall r > 0$. In other words, Eq. (13) should be held at all rounds.

$$\delta_i(r) + \sum_{\substack{j=1 \\ j \ne i}}^{n} \delta_j(r) = \delta_i(r) + (n-1)\delta_j(r) = 0 \tag{13}$$

For the sake of simplicity, we prove the correctness of Theorem 1 under two scenarios:

(1) There is no replica node (i.e., $\lambda = 1$)
(2) There are replica nodes (i.e., $\lambda > 1$)

Under the first scenario, Theorem 1 simply says the probability of being observed by the *WN* is $1/n$ for all n different *IDs*.

$$PO_i(r) = PO_j(r) = \frac{1}{n} \quad \forall\, i.\ j \text{ and } r \ge 1 \tag{14}$$

Considering $\lambda = 1$ (as is the case for the first scenario), Eq. (12) can be rewritten as below:

$$\delta_j(r) = \frac{\alpha(n-D)}{n^2} \left(PO_i(r) - PO_j(r) \right) \tag{15}$$

To show the equality presented in Eq. (14) is held in every round, we use mathematical induction. Since all elements in PO vector are initially set to $1/n$ (as discussed earlier), the base case is satisfied and $PO_i(1) = PO_j(1) = \frac{1}{n}\ \forall\, i.j$. The second induction step assumes the equality is held at the round r (i.e., $PO_i(r) = PO_j(r) = \frac{1}{n}$). Accordingly, $\delta_j(r)$ is 0:

$$\delta_j(r) = \frac{\alpha(n-D)}{n^2}\left(PO_i(r) - PO_j(r)\right) = \frac{\alpha(n-D)}{n^2}\left(\frac{1}{n} - \frac{1}{n}\right) = 0 \tag{16}$$

Thus, as is shown below, it can be concluded that $PO_j(r+1) = PO_j(r) = \frac{1}{n}$.

$$PO_j(r+1) = PO_j(r) + \delta_j(r) = PO_j(r) + 0 = \frac{1}{n} \tag{17}$$

And since $\sum_{j=1}^{n} OP_j(r+1) = 1$, it can be concluded that $PO_i(r+1) = 1/n$. Under the second scenario, Theorem 1 claims $PO_i(r) > PO_j(r)$ when $\lambda = 1$, which means $\delta_i(r) > 0$ and $\delta_j(r) < 0$ according to Eq. (13). Assume n and α are constant. Considering Eqs. (9) and (12) reveal that $\delta_i(r)$ and $\delta_j(r)$ are functions of D and λ so that, the minimum value for $\delta_i(r)$ and maximum value of $\delta_j(r)$ appears when $D = 1$ and $\lambda = 2$. In the rest of this section, we try to show $\min(\delta_i(r)) > 0$, which implicitly means $\max(\delta_j(r)) < 0$. Assuming $D = 1$ and $\lambda = 2$, $\delta_i(r)$ is written as follows:

$$\delta_i(r) = \frac{\alpha n}{(n+1)^2}\left(2 - \frac{n^2+1}{n} PO_i(r)\right) \tag{18}$$

To show $\min(\delta_i(r))$ is positive in every round r, we use mathematical induction. Considering $PO_i(1) = 1/n$, Eq. (18) is rewritten as follows to provide the base case for the induction.

$$\delta_i(1) = \frac{\alpha n}{(n+1)^2}\left(2 - \frac{n^2+1}{n} \times 1/n\right) = \frac{\alpha n}{(n+1)^2}\left(2 - \frac{n^2+1}{n^2}\right) \tag{19}$$

Since $2 > \frac{n^2+1}{n^2}$, $\delta_i(1)$ is positive and thus, the base case of the induction is satisfied. The second induction step assumes $\delta_i(r) > 0$ which means: (1) $PO_i(r+1) > PO_i(r)$; (2) $PO_i(r) < \frac{2n}{n^2+1}$. Following this assumption, we aim to show $\delta_i(r+1) > 0$ which means we should show $PO_i(r+1) < \frac{2n}{n^2+1}$. To do so, we use proof by contradiction. That is, assume $PO_i(r+1) \geq \frac{2n}{n^2+1}$. Then, from Eq. (6) we have:

$$\frac{2n}{n^2+1} \leq PO_i(r) + \frac{\alpha n}{(n+1)^2}\left(2 - \frac{n^2+1}{n} \times PO_i(r)\right) \tag{20}$$

Since from the second step, we know $PO_i(r) < \frac{2n}{n^2+1}$, the maximum of the right part of Eq. (20) is smaller than $\frac{2n}{n^2+1}$.

$$\frac{2n}{n^2+1} > PO_i(r) + \frac{\alpha n}{(n+1)^2}\left(2 - \frac{n^2+1}{n} \times PO_i(r)\right) \tag{21}$$

which contradicts with Eq. (20) and consequently contradicts the assumption of $PO_i(r+1) \geq \frac{2n}{n^2+1}$. Thus, it is concluded that $PO_i(r+1) < \frac{2n}{n^2+1}$ and $\delta_i(r+1)$ is positive.

References

[1] P. Rawat, K.D. Singh, H. Chaouchi, J.M. Bonnin, Wireless sensor networks: a survey on recent developments and potential synergies, J Supercomput 68 (1) (2014) 1–48.

[2] M. Jamshidi, M. Ranjbari, M. Esnaashari, N.N. Qader, M.R. Meybodi, Sybil node detection in mobile wireless sensor networks using observer nodes, JOIV: International Journal on Informatics Visualization 2 (May (3)) (2018) 159–165.

[3] M. Jamshidi, E. Zangeneh, M. Esnaashari, M.R. Meybodi, A lightweight algorithm for detecting mobile Sybil nodes in mobile wireless sensor networks, Comput. Electri. Eng. 64 (Nov.) (2017) 220–232.

[4] A. Diaz, P. Sanchez, Simulation of attacks for security in wireless sensor network, Sensors 16 (Nov. (11)) (2016) 1–27.

[5] K. Mishra, A.K. Turuk, A comparative analysis of node replica detection schemes in wireless sensor networks, J. Netw. Comput. Appl. 61 (Feb.) (2016) 21–32.

[6] H.R. Shaukat, F. Hashim, A. Sali, M.F. Abdul Rasid, Node replication attacks in mobile wireless sensor network: a survey, Int. J. Distrib. Sens. Netw. 10 (Dec. (12)) (2014) 1–15.

[7] Parno, A Perrig, V.D. Gligor, Distributed detection of node replication attacks in sensor networks, in: IEEE Symposium on Security and Privacy, May 2005, pp. 49–63.

[8] M. Conti, R.D. Pietro, L.V. Mancini, A. Mei, Distributed detection of clone attacks in wireless sensor networks, IEEE Trans. Dependable Secure Comput. 8 (Sep. (5)) (2011) 685–698.

[9] Y. Zeng, J. Cao, S. Zhang, S. Guo, L. Xie, Random-Walk based approach to detect clone attacks in wireless sensor networks, IEEE Journal on selected areas in communications 28 (Jun (5)) (2010) 677–691.

[10] M. Jamshidi, A.A. Shaltooki, Z.D. Zadeh, A.M. Darwesh, A dynamic id assignment mechanism to defend against node replication attack in static wireless sensor networks, JOIV 3 (1) (2019).

[11] M. Yu, C.S. Lu, S.Y. Kuo, Compressed sensing-based clone identification in sensor networks, IEEE Trans. Wireless Commun. 15 (Apr. (4)) (2016) 3071–3084.

[12] A.K. Mishra, A.K. Tripathy, A. Kumar, A.K. Turuk, A replica detection scheme based on the deviation in distance traveled sliding window for wireless sensor networks, Wireless Commun. Mobile Comput. 2017 (Jan.) (2017) 1–8.

[13] J.W. Ho, D. Liu, M. Wright, S.K. Das, Distributed detection of replica node attacks with group deployment knowledge in wireless sensor networks, Ad Hoc Netw 7 (8) (2009) 1476–1488.

[14] L Ding, Yang, M. Wu, Localization-Free detection of replica node attacks in wireless sensor networks using similarity estimation with group deployment knowledge, Sensors 17 (Jan. (1)) (2017) 160 -152.

[15] S. Roy, M.J. Nene, Prevention of node replication in wireless sensor network using received signal strength indicator, link quality indicator and packet sequence number, in: IEEE International Conference on Green Engineering and Technologies (IC-GET), 2016, pp. 1–8.

[16] J.W. Ho, M. Wright, S.K. Das, Distributed detection of mobile malicious node attacks in wireless sensor networks, Ad Hoc Netw 10 (3) (2012) 512–523.

[17] Z. Pala, K. Bicakci, M. Turk, Effects of node mobility on energy balancing in wireless networks, Comput. Electr. Eng. 41 (Jan.) (2015) 314–324.

[18] C.M. Yu, C.S. Lu, S.Y. Kuo, Mobile sensor network resilient against node replication attacks, in: IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2008, pp. 597–599.

[19] M. Safari, E. Bahmani, M. Jamshidi, S.A. Shaltooki, Advanced extremely efficient detection of replica nodes in mobile wireless sensor networks, JOIV 3 (4) (2019).

[20] J.W. Ho, M. Wright, S. Das, Fast detection of mobile replica node attacks in wireless sensor networks using sequential hypothesis testing, IEEE Trans. Mobile Comput. 10 (Jun. (6)) (2011) 767–782.

[21] M.N. Divakar, C.K. RajuTumkur, Detection of mobile replica node attacks in wireless sensor networks using sequential hypothesis testing, Network. Commun. Eng. 2 (Sep. (12)) (2010) 1–8.

[22] T. Dimitriou, E.A. Alrashed, M.H. Karaata, A. Hamdan, Imposter detection for replication attacks in mobile sensor networks, Computer Netw. 108 (Oct. 2016) 210–222.

[23] C. Zhou, Z. Wang, An two dimension detection to node replication attacks in mobile sensor networks, in: 10th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID), Sep. 2016, pp. 63–69.

[24] Z. Wang, C. Zhou, Y. Liu, Efficient hybrid detection of node replication attacks in mobile sensor networks, Mobile Inf. Syst. 2017 (Jun.) (2017) 1–13.

[25] X.M. Deng, Y. Xiong, A new protocol for the detection of node replication attacks in mobile wireless sensor networks, J Comput Sci Technol 26 (Jul. (4)) (2011) 732–743.

[26] M. Conti, R.D. Pietro, A. Spognardi, Clone wars: distributed detection of clone attacks in mobile WSNs, J Comput Syst Sci 80 (May (3)) (2014) 654–669.

[27] V. Manickavasagam, J. Padmanabhan, A mobility optimized SPRT based distributed security solution for replica node detection in mobile sensor networks, Ad Hoc Netw 37 (2016) 140–152.

[28] C.M. Yu, C.S. Lu, S.Y. Kuo, Efficient and distributed detection of node replication attacks in mobile sensor networks, in: Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th, IEEE, 2009, pp. 1–5.

[29] M. Jamshidi, S. Sheikh Abooli Poor, N. Nasih Qader, M. Esnaashari, M.R. Meybodi, A lightweight algorithm against replica node attack in mobile wireless sensor networks using learning agents, IEIE Trans. Smart Process. Comput. 8 (1) (2019) 58–70.

[30] M. Jamshidi, M. Esnaashari, A.M. Darwesh, M.R. Meybodi, Using time-location tags and WNs to defend against node replication attack in mobile wireless sensor networks, Int. J. Wireless Inf. Netw. (2019) 1–14 (in press), doi:10.1007/s10776-019-00469-0.

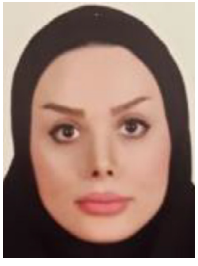
[31] Shi, A. Perrig, Designing secure sensor networks, IEEE Wireless Commun. 11 (6) (Dec. 2004) 38–43.

[32] C. Tumrongwittayapak, R. Varakulsiripunth, Detecting sinkhole attacks in wireless sensor networks, in: ICROS-SICE, Fukuoka International Congress Center, Japan, Aug. 2009, pp. 1966–1971.

[33] M.S. Yousefpoor, H. Barati, Dynamic key management algorithms in wireless sensor networks: a survey, Comput Commun 134 (2019) 52–69.

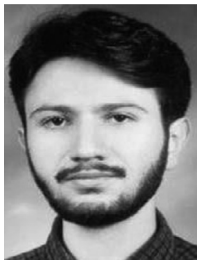
[34] K.F. Su, W.T. Wang, W.C. Chang, Detecting sybil attacks in wireless sensor networks using neighboring information, Comput. Netw. 53 (2009) 3042–3056.
[35] M. Jamshidi, E. Zangeneh, M. Esnaashari, A.M. Darwesh, M.R. Meybodi, A novel model of sybil attack in cluster-based wireless sensor networks and propose a distributed algorithm to defend it, Wireless Personal Commun. 105 (1) (2019) 145–173.
[36] M. Jamshidi, M. Ranjbari, M. Esnaashari, A.M. Darwesh, M.R. Meybodi, A new algorithm to defend against sybil attack in static wireless sensor networks using mobile observer SNs, Adhoc & Sensor Wireless Netw. 43 (2019) 213–238.
[37] M. Jamshidi, E. Zangeneh, M. Esnaashari, A.M. Darwesh, M.R. Meybodi, A novel model of sybil attack in cluster-based wireless sensor networks and propose a distributed algorithm to defend it, Wireless Personal Commun. 105 (1) (2019) 145–173.
[38] A. Sobeih, J.C. Hou, L.C. Kung, N. Li, H. Zhang, W.P. Chen, H.Y. Tyan, H. Lim, J-Sim: a simulation and emulation environment for wireless sensor networks, IEEE Wireless Commun. 13 (4) (2006) 104–119.

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 .



Shokooh Sheikh Abooli Poor received the B.S. degree in Computer Engineering from the Islamic Azad University, Ramhormoz, Iran, in 2013, and M.S. degree in Computer Engineering from Islamic Azad University, Broujerd, Iran, in 2017. Her research interests include wireless networks, learning systems, security, data mining, and recommender systems.

Abbas Arghavani received his BSc degree in Computer Engineering from University of Payame Noor, Iran, 2008; his MSc degree in Information Technology from Qazvin Islamic Azad University, Iran, 2012; and his PhD in Computer Science from the University of Otago, 2019. He has served as the reviewer of high-ranked Journals and Conferences. His research interests include wireless sensor networks, body networks, wireless signal processing, reinforcement learning, network security, and game theory.



Mehdi Esnaashari received the B.S., M.S. and Ph.D. degrees in Computer Engineering all from the Amirkabir University of Technology in Iran, in 2002, 2005, and 2011 respectively. He worked at Iran Telecommunications Research Center as an Assistant Professor from 2012 to 2016. Currently, he is an Assistant Professor in Computer Faculty of K. N. Toosi University of Technology. His research interests include computer networks, learning systems, soft computing, and information retrieval.

Abdusalam Abdulla Shaltooki received the B.S. degree in Computer Engineering from University of Kharazmi in Iran, in 2004, and M.S. degree from the University of Sulaimani in the Kurdistan Region of in 2009. He worked from 2005 to 2010 as a lab Assistance and Instructor at the University of Sulaimani. Currently, he has a full time job at the University of Human Development. His research interests include computer networks, cloud system, and data mining.

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.