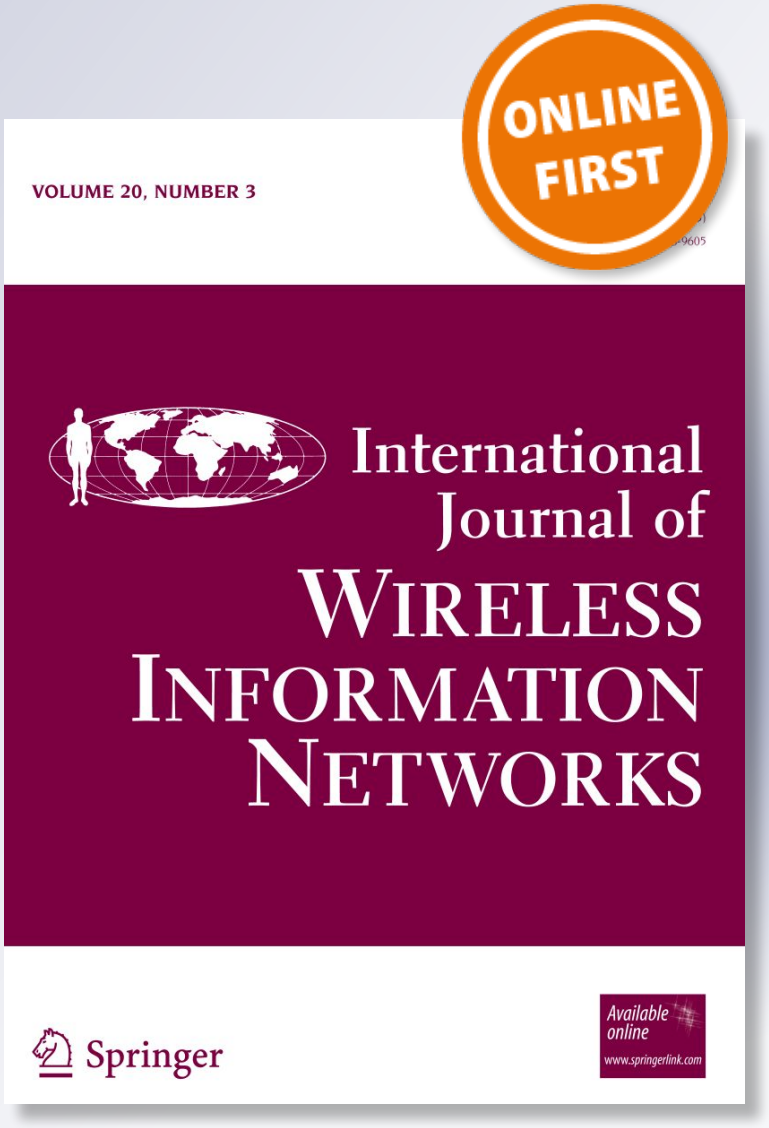
Using Time-Location Tags and Watchdog Nodes to Defend Against Node Replication Attack in Mobile Wireless Sensor Networks

Mojtaba Jamshidi, Mehdi Esnaashari, Aso Mohammad Darwesh & Mohammad Reza Meybodi

International Journal of Wireless Information Networks

ISSN 1068-9605

Int J Wireless Inf Networks
DOI 10.1007/s10776-019-00469-0



Springer

International Journal of Wireless Information Networks
https://doi.org/10.1007/s10776-019-00469-0

Using Time-Location Tags and Watchdog Nodes to Defend Against Node Replication Attack in Mobile Wireless Sensor Networks

Mojtaba Jamshidi[1] · Mehdi Esnaashari[2] · Aso Mohammad Darwesh[1] · Mohammad Reza Meybodi[3]

Received: 11 May 2018 / Revised: 8 May 2019 / Accepted: 12 October 2019

Abstract
Node replication attack is one of the well-known and dangerous attacks against Wireless Sensor Networks (WSNs) in which adversary enters the network, searches randomly and captures one or multiple normal nodes. Adversary extracts data and keying materials of the captured node and generates several copies of that node and deploys them in the network. In this paper, a novel algorithm using watchdog nodes is proposed to detect replica nodes in mobile WSNs. The main idea of the proposed algorithm is inspired by maximum predefined speed for sensor nodes and using time-location tags by the watchdog nodes to detect replica nodes. Watchdog nodes collaborate to measure sensor nodes' speed in the environment and if they find that a node moves faster than a predefined threshold, they mark it as a malicious node, because such replica node in different regions of the network is moving faster than usual in different regions of the network. The proposed algorithm is implemented by J-SIM simulator and its performance is evaluated in terms of false detection and true detection rates through some experiments. Experiment results show that the proposed algorithm is able to detect 100% of replica nodes, while the false detection rate is less than 0.5%.

Keywords Mobile Wireless Sensor Network · Replication attack · Replica nodes · Time-location tags · Watchdog nodes

1 Introduction

WSN is used mainly in environments which are difficult for a human to work within them or sometimes it is dangerous. Most of WSN applications are in the military and health industry. Usually, there are hundreds and thousands of small and cheap sensor nodes in each sensor network. Considering sensor nodes' limitations in terms of energy, memory and computational power together with broadcast nature of wireless communication and being non-resistant against adversary tamper, establishing security in such networks is an important issue [1–3].

WSN may be attacked in different ways. Replication of nodes is the most dangerous one [4, 5]. Distributing a huge number of nodes in the network is one of the challenges that can be used to capture normal nodes and extract sensible information that can be used to duplicate these nodes. Duplicated nodes (named replica nodes) contain an identification (ID) and keying materials of the captured nodes. In this way, they can establish a shared key with other normal nodes of the network. When the shared key is established, these adversary nodes also can react as normal nodes, because they have ID and keying materials. Thus, they can attack the network and deploy other nodes in the network.

Replica nodes can create sharable keys with normal nodes and the base station easily because actual protocols using in securing communication allow it. And adversary nodes can detect a part of the network's traffic passing through duplicated nodes and inject distorted data to destroy the mission

✉ Mojtaba Jamshidi
jamshidi.mojtaba@gmail.com

Mehdi Esnaashari
esnaashari@kntu.ac.ir

Aso Mohammad Darwesh
aso.darwesh@uhd.edu.iq

Mohammad Reza Meybodi
mmeybodi@aut.ac.ir

[1] Department of Information Technology, University of Human Development, Sulaymaniyah, Iraq

[2] Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran

[3] Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran

Published online: 26 October 2019

of the nodes, also disrupt common protocols of the network including clustering and data aggregation [4, 5].

Replica nod attacks in static WSNs have been the subject of many researchers and a number of algorithms are proposed [5–14]. But these algorithms are depending on GPS or transmitting location claim of nodes or they are specific to some particular topologies, like Grid, thus they cannot be used in mobile WSN Since the nodes move continuously [15, 16]. On the other hand, [17–26] suggest algorithms for replica node attacks in mobile WSNs. These algorithms have some drawbacks because of having a complicated process, the low rate in detecting replica nodes, having a high communication and memory overhead, instability and using public key and digital signature to determine nodes location. The main idea of all these algorithms is discussed in the next sections.

In our previous works [27, 28], we used watchdog nodes and a distributed bitwise-labeling mechanism based on nodes' movement behaviors to detect Sybil nodes in mobile WSNs. In this paper, a novel algorithm using watchdog nodes is proposed to detect replica nodes in mobile sensor networks. In the proposed algorithm, there are a number of watchdog nodes which detect replica nodes through collaboration and walking in the network. After facing sensor nodes, watchdog nodes record their time-location tag in their buffer. Then, they collaborate with each other to detect the replica nodes considering the contents of the buffers.

The rest of this paper is organized as follows. In Sect. 2, previous studies are reviewed. Section 3 explains the system model and assumptions. Section 4 describes the proposed algorithm. Performance is evaluated in Sect. 5 and simulation results are also presented in this section. The last section concludes the paper.

2 Related Work

In this section, we discuss the existing algorithms against replica node attack in static and mobile WSNs, separately.

2.1 Algorithms in Static WSNs

There are many algorithms such as Line-Selected Multicast (LSM) [5], Randomized, Efficient and Distributed (RED) [6, 7], and Distributed Deterministic and Resilient (DDR) [10] which are based on transmitting location claims to some witness locations/nodes of the network. When any witness receives more than one different location claim for a particular node, such as u, marks u as a replica node.

SET [8] is another algorithm to detect replica nodes in static WSNs which uses set operations (union and intersection) on some parts of the network. Single Deterministic Cell (SDC) and Parallel Multiple Probabilistic Cells (P-MPC) [9] were proposed to detect replica nodes on a grid network. These algorithms use localized multicast and location claims.

Random Walk (RAWL) [11] were proposed based on witness nodes and a random walk to detect replica nodes. In this algorithm, for each node u, several random walks are taken, and nodes that pass are selected as witness nodes of u. Table-assisted Random Walk (TRAWL) [11] uses a trace table to reduce the memory cost of the RAWL. Also, Yu et al. [12] employed a compressed sensing method to detects replica nodes in the static WSNs.

Four another location claims-based algorithms ((B-MEM, BC-MEM, C-MEM, and CC-MEM)) [13] are proposed which use Bloom filters [29] to compress the information stored at the sensors, and use two techniques, called cell forwarding and cross forwarding, to improve detection probability and reduce memory consumption. Jamshidi et al. [14] proposed an algorithm based on a dynamic ID–assignment mechanism to defend against a replica node attack in the static WSNs. This algorithm uses a multi-tree architecture based on a multi-sink architecture for dynamic assignment of IDs to the sensor nodes after deployment in a network environment. If this mechanism is used, replica nodes generated by the adversary cannot be easily attached to the network.

2.2 Algorithms in Mobile WSNs

Two algorithms EXtremely Efficient Detection (XED) [17] and Dimitriou et al. [24] were proposed based on exchanging random numbers between each pair nodes u and v when they meet each other during the network lifetime. The main idea of these algorithms is that if node u meets another node, like v, at time T_1, it sends a random number, r, to v. When nodes u and v meet each other once again at time T_2, u asks v for the random number sent to it at time T_1, and expects v to send it 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. Disadvantages of these algorithms are a slow detection process and high overhead in communications and memory. Each node needs keep $n-1$ (n is the total number of nodes in the network) random numbers and should send/request/verify d (the number of neighbors on average) random numbers in each round of the network lifetime.

Computing node's movement speed is considered as another technique used to detect replica nodes in the mobile WSNs. Legal node speed should not exceed the maximum speed of the configured system. Algorithms [18–20] employ this technique to detect replica nodes. Replica nodes of u make other nodes think that node u moves faster than a predefined speed. In this case, they can mark u as a replica node. Disadvantages of these algorithms are high memory and communication overheads, being expensive because of

using GPS for all nodes in the network. Also, these algorithms are centralized and suffer from single-point failure.

A pair-wise key protocol based on polynomials and Bloom filters [21] is proposed to detect replica nodes in the mobile WSNs. This algorithm employs the pair-wise keys to guarantee that replica nodes cannot be located close to nodes with the real identities. Disadvantages of this algorithm are being centralized, not being scalable, long detection process, and low flexibility when injecting new nodes to the network.

Time Domain Detection (TDD) and Spatial Domain Detection (SDD) [22] are proposed to detect replica node attack in mobile ad hoc networks. They use a cryptographic one-way hash function to force replicas to keep on generating paradoxes whose detection reveals the existence of replica attacks in the network. These algorithms impose a high overhead, especially in terms of processing, on resource-constraints sensor nodes.

Zhou and Wang [23] proposed a scheme to defend against replication attack in mobile WSNs. This scheme has two levels, local detection, and global detection. The local detection is performed in a local area much smaller than the whole deployed area to improve the meeting probability of the contradictory locations and the global detection in a longer time period epoch verity location claim with every node it meets. One of the disadvantages of this algorithm is high memory overhead because it assumed that each node has enough storage space to store the last h rounds track neighbor information.

Conti et al. [25] proposed two protocols to defend against replication attack, called History Information-exchange Protocol (HIP) and its optimized version (HOP). Both of these protocols employ local information and node mobility to detect replica nodes. These protocols executed in rounds. For each protocol round, each sensor stores a log of the neighbors met during that round, with their locations. Two protocols differ in the amount of computation required. Disadvantages of these algorithms are high memory and communication overheads, being expensive because of using GPS for all nodes in the network.

Manickavasagam and Padmanabhan [26] presented a distributed algorithm to detect replica nodes in the 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 is based on the fact that when several replicas transmit simultaneously, multiple messages with the same source ID but physically different sources are alive in the network. Disadvantages of this algorithm are high processing and memory overheads. Each node needs to verify many packets in each round of the algorithm execution.

Generally, the above algorithms have some drawbacks because of having a complicated process, a high cost because of using many GPSs, the low rate in detecting replica nodes, having a high communication/memory/processing overhead, and lack of scalability because of being centralized. The proposed algorithm aims to alleviate the drawbacks of these algorithms as much as possible.

3 System Assumptions and Attack Model

In the network, there are two kinds of nodes, sensor nodes, and watchdog nodes. Each node has a unique ID. These two sets of nodes are randomly distributed in two-dimensional environments. The role of each of them is different, sensor nodes take the overall mission of the network like collecting information and sharing it with the base station. Watchdogs on the other side take the responsibility of detecting replica nodes.

All nodes move in the network according to the random walk mobility model [30]. In this mobility model, each node chooses a random direction (uniformly distributed in $[0,\ 2\pi]$) and random speed (also uniformly distributed in $[v_{min}, v_{max}]$); it then moves for a time period (or over a fixed distance) with this speed, then it repeats its choice.

The position is very important for watchdog nodes which have GPS as well, but not for sensor nodes. We consider the same radio range r for all nodes and all nodes are mobile and move randomly as waypoint throughout the network's lifetime. The communication among nodes is done through a wireless radio channel and employs omnidirectional dissemination.

By default, sensor nodes are not tamper-resistant and the adversary can capture its secret information and reprogram them. While we assume that watchdog nodes are tamper resistant and cannot be decrypted [31, 32]. Each sensor node is equipped with a tamper-resistant component for the storage of sensitive e.g. key data, thus limiting the damage following the capture of nodes.

Nodes should broadcast a "Hello" message, a request route, data sending, or keep alive message periodically, after a defined time period t or changing location because all nodes are mobile [27, 28, 33]. Nodes in mobile WSN need this exchanges because it is one of their requirements. So, each node can detect its current neighbors, establish security keys with them when needed, and communicate with them and create its routing table. Concerning watchdog nodes, they don't send such messages to remain hidden in the network, because they are responsible for detecting malicious nodes (replica nodes).

It is assumed that the nodes use a mobile-specific routing protocol, such as [34], to communicate with the base station. Also, considering the nature of wireless communications in WSNs, the adversary can launch an eavesdropping attack to discovers the information transmitted in the network's links. Applying key pre-distribution schemes is one of the effective

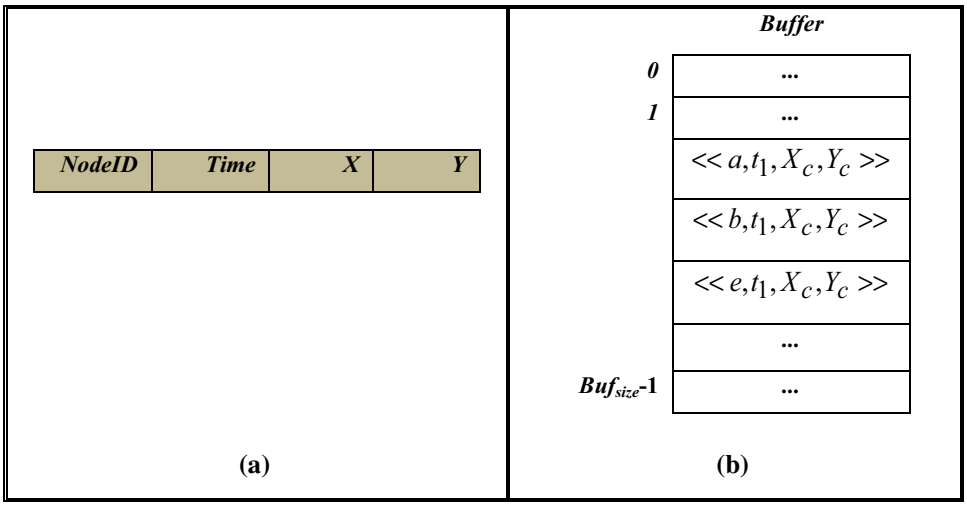


Fig. 1 **a** Structure of time-location tags, and **b** storing time-location tags in watchdog nodes' buffer

and promising mechanisms to protect against such threats [35]. Therefore, it is assumed that watchdog nodes use a group key management protocol, like scheme [36], to secure their communication (exchanging their buffer content).

Watchdog nodes are aware of themselves and they can communicate when meeting each other. Each watchdog node has a list of watchdog nodes' ID. Also, we assumed that the sensor network is developed in an adversary environment, thus, in such insecure network, an adversary can capture nodes, create copies of them, and inject them into the network.

We also assumed that each replica node, like normal nodes, in each time period t, broadcasts a "Hello", routing request, or transmitting data. An adversary might deploy replica nodes in specific locations, or they could be mobile like normal nodes. None of these states affects the proposed algorithm.

Since the proposed algorithm is only executed by the watchdog nodes, the sensor nodes don't need time synchronization. Also, the watchdog nodes monitor the sensor nodes movements independently and begin to communicate when they meet each other. Thus, it is not necessary to synchronize watchdog nodes.

4 The Proposed Algorithm

The main idea of the proposed algorithm is to use time-location tags by the watchdog nodes to detect replica nodes. In the following, the proposed algorithm is described in details.

In the proposed algorithm, each watchdog node has a buffer to record time-location tags. Structure of time-location tags is given in Fig. 1a. Node identity, time, and location coordinate are stored in *NodeID*, *Time*, *X* and *Y* fields, respectively. The capacity of the watchdog nodes' buffer is limited and equal to Buf_{size}. Structure of the buffer which is used by the watchdog nodes is circular. That is, each watchdog node keeps at most Buf_{size} recent time-location tags in its buffer.

After being deployed in the network, nodes move periodically and send messages (data, route request, Hello, being alive and etc.). For example, each node sends a Hello message after each t time units or whenever it arrives a new location of the network. Thus, nodes will be aware of their current neighbors regarding such messages. Each watchdog node W_i records a time-location tag about node u in its buffer when facing each node u. assuming that the watchdog node W_i is located in (X_c, Y_c) at time t_1 and nodes a, b, and e are at its neighborhood, watchdog node W_i records three different time-location tags in its buffer according to Fig. 1b.

It should be noted that in the proposed algorithm, only watchdog nodes are aware of their location and other sensor nodes do not require determining their location. Thus, in the proposed algorithm, no overhead or cost is imposed for locating sensor nodes. As can be seen in Fig. 1b, W_i records its current location, (X_c, Y_c), in time-location tag of nodes a, b and c.

In the proposed algorithm, watchdog nodes collaborate to identify replica nodes. When a watchdog node W_i meets at least one other watchdog node in its neighborhood, it encrypts its buffer content by a shared group key [36] and broadcasts it to the neighbor watchdog nodes. Then, watchdog nodes use the contents of these buffers to make the decision about the nature of sensor nodes. Following, the detection process is described in details.

In order to detect replica nodes, if two watchdog nodes W_i and W_j meet each other at time t_k, they send their buffer to each other. Then each one detects replica nodes by investigating the buffer contents. For instance, consider Fig. 2 and assume that the adversary has injected two replica nodes of u in the network. Now, assume that these two copies of nodes u have appeared at the neighborhood of W_i at time t_1 (at X_i

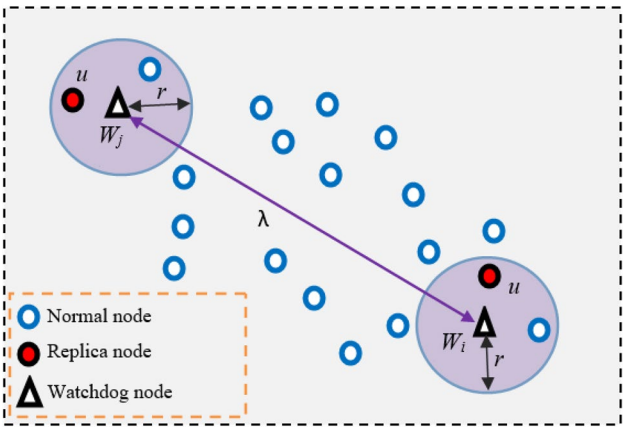


Fig. 2 A probable topology of the network for detecting replica node

and Y_i) and at the neighborhood of W_j at time t_2 (at X_j and Y_j). In this case, nodes W_i and W_j record time-location tags of these nodes in their buffer.

Now, assume that at time t_k ($t_k > t_1, t_2$), watchdog nodes W_i and W_j meet each other. Thus, these watchdog nodes send their buffer content to each other. Then, one of these two watchdog nodes (for example, the watchdog node which has the lower ID) scans its buffer content and compare it with the received buffer from the other watchdog node. Since two time-location tags may found for the node u (tag $\ll u, t_1, X_i, Y_i \gg$ in the buffer of W_i and tag $\ll u, t_2, X_j, Y_j >$ in buffer of W_j), if Eq. (1) is satisfied, u is marked as replica node.

$$dist(X_i, Y_i, X_j, Y_j)/Max\_speed > |t_1 - t_2| \quad (1)$$

Here, *dist* is the Euclidean distance between two points and *Max_speed* is a threshold (maximum predefined speed of nodes). This decision is because if Eq. (1) is satisfied, u moves faster than *Max_speed* which is contrary to maximum predefined speed for nodes. On the other hand, the existence of replica nodes u in the network make others think that node u moves faster than the predefined speed since it is in many locations of the network in the same time. Thus, the proposed algorithm can detect replica nodes easily.

Indeed, time-location tag of the node u might not be in the buffer of one of these watchdog nodes at time t_k when they meet each other. This occurs when the content of the buffer is rewritten with new elements. In this situation, watchdog nodes, W_i *and* W_j will not be able to detect replica node u at time t_k. The maximum speed of nodes, total number of nodes in the network and buffer's capacity affect this situation. The smaller is the capacity of the buffer, the lower is the probability of the time-location tags to be available in both watchdog nodes' buffers at time t_k. Moreover, since the total number of nodes in the network increases, the average number of neighbors of each node increases, thus watchdog nodes insert more tags in their buffer at each random walk (arriving a new location of the network or after a time period t). Therefore, the lifetime of a time-location tag in the watchdog's buffer decreases and it is overwritten with new tags. Similarly, as the speed of nodes increases, the probability that time-location tag of node u exists in both watchdog nodes' buffer increases because if two watchdog nodes are far from each other, they meet each other after a short time and a few executions of the algorithm. Experiment results verify this.

Therefore, the proposed algorithm includes two parallel phases: (1) recording information in watchdog nodes' buffers and (2) run detection procedure when two watchdog nodes meet each other. In each round of executing the proposed algorithm, time-location tags are recorded in the watchdog nodes' buffers, but detection procedure might run or not because detection procedure is run only when watchdog nodes meet each other.

Each watchdog node upon detecting a replica node, for example, node ID u, sends a message to the base station so that the network manager is informed that the network has been attacked to undertake stricter security actions. The base station may propagate an alarm message to the whole network in order to nodes avoid any communication with the node with ID u.

5 Performance Evaluation and Simulation Results

In this section, first the proposed algorithm is evaluated in terms of memory, communication and computation overhead and the results are compared with other existing algorithms. Then, we have investigated the security analysis of the proposed algorithm. At the end, we presented the simulation results.

5.1 Performance Evaluation

5.1.1 Memory Overhead

Since the proposed algorithm is executed only by watchdog nodes, no memory overhead is imposed on sensor nodes. But each watchdog node requires a Buf_{size} in memory to store the time-location tags in its buffer. It should be noted that the number of watchdog nodes in the network is much less than the number of sensor nodes.

The minimum Buf_{size} can be selected in one of the following ways:

- The buffer size is considered as $BufSize = d * L$ where d is the average number of neighbors of a node (network density) and L is the network's diameter in terms of the

Table 1 Performance comparison of the proposed algorithm and the other algorithms in terms of memory, communication, and computation overhead

Algorithm	Network type	Algorithm type	Memory overhead (per network)	Communication overhead (per network)	Computation overhead (per node)	Technique
LSM [5]	Stationary	Distributed	$O(n\sqrt{n})$	$O(n\sqrt{n})$	$O(\sqrt{n})$	Location claim based
RED [7]	Stationary	Partially distributed	$O(n \times d)$	$O(n\sqrt{n})$	$O(\sqrt{n})$	Location claim based
RAWL [11]	Stationary	Distributed	$O(n \log n\sqrt{n})$	$O(n \log n\sqrt{n})$	–	Location claim based
XED [17]	Mobile	Distributed	$O(n \times d)$	$O(n \times d \times R)$	$O(R \times d)$	Random number and node meeting based
SPRT [19]	Mobile	Centralized	$O(n^2)$	$O(n\sqrt{n})$	$O(R \times n \times b \times p)$	Speed and node meeting based
Algorithm [21]	Mobile	Centralized	$O(n \times d)$	$O(n \times \log n)$	$O(R \times n)$	Key-based
SDD [22]	Mobile	Distributed	$O(n^2)$	$O(n \times d)$	$O(d)$	One-way hash function
HOP [25]	Mobile	Distributed	$O(nRd^2 + nRd)$	$O(nRd^2)$	$O(Rd^3)$	Location-based and node meeting based
Algorithm [26]	Mobile	Distributed	$O(n^2)$	$O(nR)$	$O(R\sqrt{n})$	Sequential probability ratio test
The proposed algorithm	Mobile	Partially distributed	$O(w \times n)$	$O(R \times w)$	$0 \sim O(R \times n^2)$	Node meeting based

n: the number of total nodes in the network

d: the average number of neighbors of a node

$b \times p$: the number of location claims which are sent by every node to the base station

R: the number of monitoring rounds that are carried out by an algorithm

w: the number of watchdog nodes in the network

Hint: w is much less than n and is less than d

Hint: the computation overhead of the proposed algorithm is zero to the sensor nodes and is $O(R \times n^2)$ to the watchdog nodes

hop. Parameter L is calculated as $L = \left\lceil \frac{Max(diameter)}{r} \right\rceil$ where M$ax(diameter)$ is the largest diameter of the network and r is the radio range of the nodes.

- The buffer size is selected as $BufSize = \frac{n}{w}$ so that it is assumed that the total capacity of all watchdog nodes' buffer is equal to the total number of nodes in the network (means, n). In this case, time-location tags about all nodes would exist in watchdog nodes' buffer most of the times. This selection is easier and more efficient.

And, the maximum Buf_{size} can be considered n to increase the probability of keeping tags for all nodes by each watchdog node. In this situation, the proposed algorithm is also able to detect replica nodes, even if the adversary clones all sensor nodes.

Therefore, we can consider that the memory overhead of each watchdog node is on the order of $O(n)$. Table 1 compares the memory overhead (per network) of the proposed algorithm with other existing algorithms.

5.1.2 Communication Overhead

Since the proposed algorithm only employs Hello messages, route request, data transmission and being alive messages, and these messages are requirements of the mobile sensor networks, no communication overhead is imposed on sensor nodes. But watchdog nodes broadcast their buffer contents to each other whenever they meet. For each watchdog node W_i, the possibility of meeting another watchdog node, like W_j, in each movement or monitoring round, is $\left(\frac{1}{n-1} \times d\right)$. Here, n is the number of total nodes in the network and d is the average number of neighbors for each 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. (2)

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

Thus the probability of meeting (P_m) at least one another watchdog node by W_i, in each round of the monitoring phase, can be calculated with Eq. (3):

$$P_m = 1 - \frac{\binom{n-w}{d}}{\binom{n-1}{d}} \tag{3}$$

where w is the number of watchdog nodes in the network.

Each watchdog node W_i should broadcast its buffer content to other watchdog nodes if it meets at least one of them. Hence, the communication overhead of the proposed algorithm is $w \times P_m$ and is on the order of $O(w)$, in each monitoring round. Therefore, the communication overhead of the proposed algorithm is on the order of $O(R \times w)$, for R rounds of monitoring. Table 1 also compares the communication overhead (per network) of the proposed algorithm with other existing algorithms.

5.1.3 Computation Overhead

Since the proposed algorithm is executed by watchdog nodes, thus no computation overhead is imposed on sensor nodes. But each watchdog node W_i after meeting another watchdog node, such as W_j, needs to execute the detection procedure. For this purpose, watchdog node W_i should compare all time-location tags of its buffer with what it has received from W_j. Therefore, the computation overhead of the proposed algorithm for each watchdog whenever it meets another watchdog is on the of order $O(n^2)$. Table 1 also compares the memory overhead (per node) of the proposed algorithm with other existing algorithms.

In summary, the algorithm [21] is the main competitor of the proposed algorithm. The memory overhead of the proposed algorithm is on the order of $O(w \times n)$ while this overhead is on the of $O(n \times d)$ for algorithm [21]. The number of watchdog nodes (w) in the proposed algorithm follows from Eq. (6) which is usually less than the number of neighbors (d) which follows from Eq. (2), especially in the dense networks. Hence, the proposed algorithm is superior in terms of memory consumption.

In terms of computation, both algorithms impose almost the same overhead to the network. The proposed algorithm imposes zero and $O(R \times n^2)$ computation overhead to the sensor nodes and watchdog nodes, respectively while algorithm [21] imposes $O(R \times n)$ computation overhead to every node in the network.

Also, in terms of communication, the proposed algorithm imposes $O(R \times w)$ overhead to the total nodes in the network but this overhead in the algorithm [21] is $O(n \times \log n)$. It is clear that the communication overhead of algorithm [21] is less than the communication overhead of the proposed algorithm, for very sparse networks, but the proposed algorithm imposes much less communication overhead compared to algorithm [21], for dense networks.

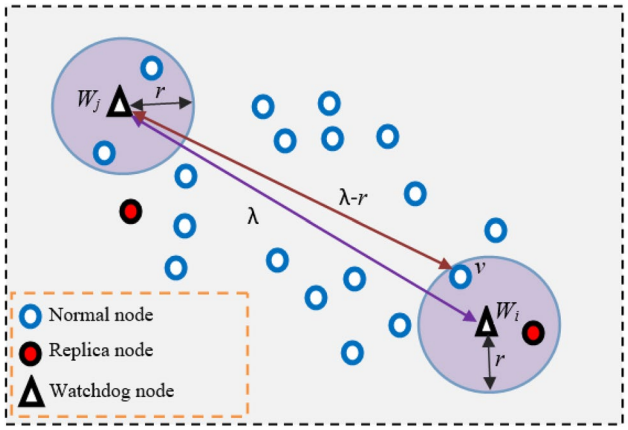


Fig. 3 A probable topology of the network at time t_1

In addition, it should be noted that the algorithm [21] is centralized which make it non-scalable and suffering single point failure. But the proposed algorithm is partially distributed which make it more scalable as well as a failure of a watchdog does not break the entire algorithm.

5.2 Security Analysis

5.2.1 False Detection

The proposed algorithm seldom detects replica nodes incorrectly. Here we presented a possible case in which the proposed algorithm detects some normal nodes as replica node incorrectly.

In Fig. 3, assume that distance between watchdog nodes W_i *and* W_j at time t_1 is λ and non-replica node v is in the neighborhood of W_i and its distance from W_j is $\lambda - r$.

At this time, a time-location tag about node v is recorded in the buffer of W_i (at location X_i and Y_i). Now, if nodes v and W_j move towards each other with *Max_speed* after t_1, both nodes might be in each other's neighborhood after Δ_t where is obtained as Eq. (4):

$$\Delta_t = t_1 + \frac{\lambda - (2 \times r)}{2 \times Max\_speed} \tag{4}$$

In this equation, $\lambda - (2 \times r)$ is the minimum distance that these nodes should pass to be in each other's neighborhood and since it is assumed that these two nodes move towards each other with maximum possible speed simultaneously, this distance is passed with double speed, as illustrated in Fig. 4.

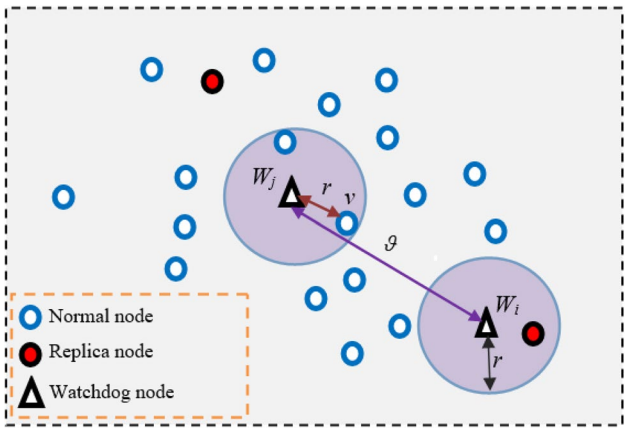


Fig. 4 A probable topology of the network at time Δ_t

Now, at time Δ_t, watchdog node W_j (at location X_j and Y_j) record a time-location tag about node v in its buffer, and since it has been assumed that nodes v and W_i move towards each other and with maximum speed after t_1, when these nodes meet each other, the distance of W_j (at location X_j and Y_j) at time Δ_t and W_i (at location X_i and Y_i) at time t_1 would be ϑ which is obtained from Eq. (5):

$$\begin{aligned} &\vartheta = dist(X_i, X_i, X_j, Y_j) = \lambda - \frac{\lambda - (2 \times r)}{2} \\ &\Rightarrow \vartheta = \frac{\lambda + (2 \times r)}{2} \end{aligned} \tag{5}$$

Now if two watchdog nodes W_i *and* W_j meet each other at time t_k ($t_k > \Delta t$), applying Eq. (1) to two time-location tags, provided that time-location tag still exists in the buffer of both watchdog nodes, we have:

$$\begin{aligned} &dist(X_i, Y_i, X_j, Y_j)/Max\_speed > \left|t_1 - \Delta_t)\right| \\ &\Rightarrow \frac{\vartheta}{Max\_speed} > \left|t_1 - \Delta_t)\right| \\ &\Rightarrow \frac{(\frac{\lambda+(2\times r)}{2})}{Max\_speed} > \frac{\lambda - (2 \times r)}{2 \times Max\_speed} \\ &\Rightarrow \frac{\lambda + (2 \times r)}{2 \times Max\_speed} > \frac{\lambda - (2 \times r)}{2 \times Max\_speed} \\ &\Rightarrow (2 \times r) > -(2 \times r) \end{aligned}$$

Since the above relation is satisfied, watchdog nodes W_i *and* W_j mark legal node v as a replica node. Indeed, this situation occurs rarely and clearly depends on the number of watchdog nodes, maximum predefined speed and execution time of the proposed algorithm. Increasing each of these three parameters causes false detection. Experiments results verify this.

5.2.2 Watchdog Selection

Choosing the number of watchdog nodes required in the proposed algorithm can be a challenge. The dynamic nature of the mobile WSNs has made it impossible to determine the exact number of observer nodes needed in the network. Certainly, at least two watchdog nodes are required. But, the maximum number of required watchdog nodes can be estimated in an ideal case in which the whole network environment is covered by watchdog nodes. Hence, the minimum and the maximum number of required watchdog nodes can be calculated with Eq. (6):

$$Watchdog\# \begin{cases} 2 & \min \\ \left\lceil \frac{X \times Y}{r^2 \times \pi} \right\rceil & \max \end{cases} \tag{6}$$

5.3 Simulation Results

The proposed algorithm is implemented by J-SIM simulator [37] and its performance is evaluated in terms of false detection and true detection rates. Moreover, the efficiency of the proposed algorithm is compared with the algorithm presented in [7, 13, 21].

- *True Detection Rate* percentage of replica nodes which are detected correctly.
- *False Detection Rate* percentage of legal nodes (normal or non-replica) which are incorrectly detected as replica nodes.

In simulations, it is assumed that the total number of nodes in the network is n which are distributed in a $100 * 100$ m^2 region. The number of watchdog nodes is also considered w (w nodes picked from n nodes). Adversary captures M legal nodes in the network and creates R replicas from each and injects them to the network. Radio range of all nodes is $r = 15$ meters (except in experiments 6 and 7). In all simulations, random walk mobility model is used. In all experiments except experiments 3, and 7, the maximum speed of each node is considered as $Max\_speed = 10$ m/s. In addition, it is assumed that every second, each node appears in the network by sending a message (Hello, data, route request or etc.). That is, the monitoring phase of the proposed algorithm is repeated every second. In order to verify the results, each simulation is repeated 50 times and The final result is obtained by averaging these 50 results.

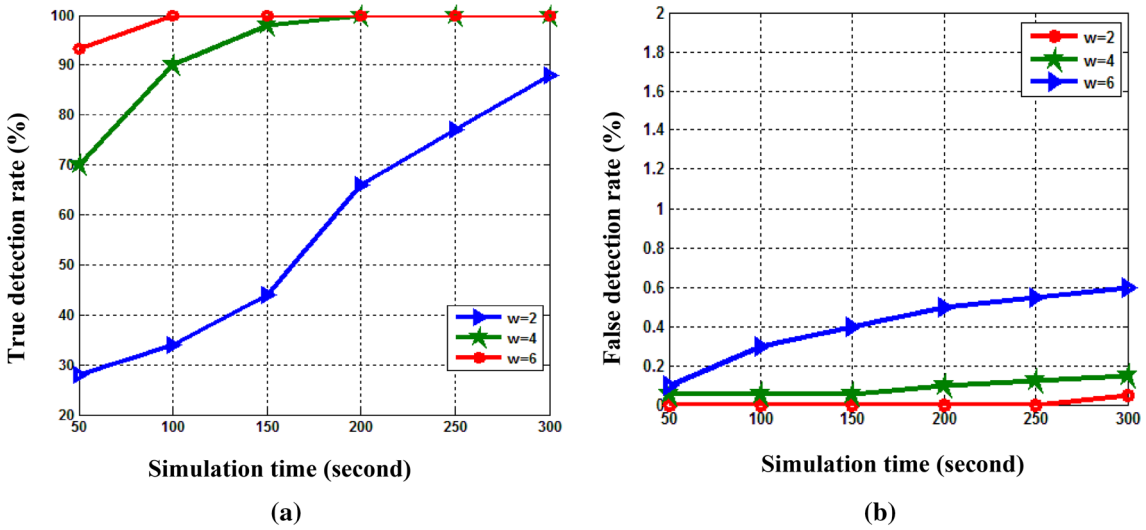


Fig. 5 Effect of the number of watchdog nodes, w, on the true detection rate (**a**) and the false detection rate (**b**) of the proposed algorithm

5.3.1 Experiment 1

the purpose of this experiment is to evaluate the effect of the number of watchdog nodes, w, on the efficiency of the proposed algorithm. In this experiment, parameters are set as $n = 200$, $M = 5$, $R = 5$, $Buf_{size} = 100$, $Max\_speed = 10\,\text{m/s}$ and the number of watchdog nodes varies between 2 to 6 ($w = 2, 4, 6$) and efficiency of the proposed algorithm is evaluated. Figure 5 shows the results of this experiment in terms of both true and false detection rates.

The results of this experiment presented in Fig. 5a show that by increasing simulation time (or network's lifetime) true detection rate increases, because by increasing simulation time, number of repetitions of the proposed algorithm also increases, thus the probability of detecting replica nodes by the proposed algorithm increases. Also, the results of this experiment show that by increasing the number of watchdog nodes in the network, the true detection rate increases. The reason is clear. By increasing number of watchdog nodes in the network, more regions are covered by the watchdog nodes, therefore replica nodes will meet watchdog nodes with higher probability. Thus watchdog nodes detect replica nodes much faster. As can be seen in Fig. 5a, true detection rate of the proposed algorithm, when there are only two watchdog nodes in the network is 66% after 200 s, and when there are four watchdog nodes in the network, it reaches 100% after 150 s and when there are six watchdog nodes in the network, it reaches 100% after 100 s.

Furthermore, the results of this experiment presented in Fig. 5b show that increasing simulation time or increasing number of watchdog nodes in the network, increases false detection rate of the proposed algorithm, because increasing any of these parameters, simulation time and number of watchdog nodes, create a situation mentioned in Sect. 5.2 and it was proved that it results in more false detections of the replica nodes. As the results presented in Fig. 5b show, the false detection rate of the proposed algorithm when there are only two watchdog nodes in the network is 0%, when there are four or six watchdog nodes in the network, this rate is less than 0.4% and 0.6% respectively (less than 1%).

5.3.2 Experiment 2

The purpose of this experiment is to investigate the effect of watchdog nodes' buffer capacity on the performance of the proposed algorithm. In this experiment, parameters are set as $n = 200$, $M = 5$, $R = 5$, $w = 4$, $Max\_speed = 10\,\text{m/s}$ and capacity of buffers varies from 50 to 150 ($Buf_{size} = 50$, 100, 150) and the effect of these variations have been investigated at different simulation times. Figure 6 shows the results of this experiment in terms of both true and false detection rates.

Figure 6a shows that by increasing capacity of buffers, true detection rate of the proposed algorithm increases, because by increasing capacity of watchdog nodes' buffer, time-location tags are kept in the buffer more times, thus when two watchdog nodes meet each other, probability that a replica node exists in both watchdogs' buffers increases. Thus the watchdog nodes would be able to detect the replica node. Experiment results also verify this. It can be seen in Fig. 6a that after 50 s, the true detection rate of the proposed algorithm for $Buf_{size} = 50$, $Buf_{size} = 100$, and $Buf_{size} = 150$ is 36%, 70%, and 90% and after 100 s of simulation these value would be 46%, 90%, and 100%.

Furthermore, the results of this experiment in Fig. 6b show that by increasing capacity of watchdog nodes'

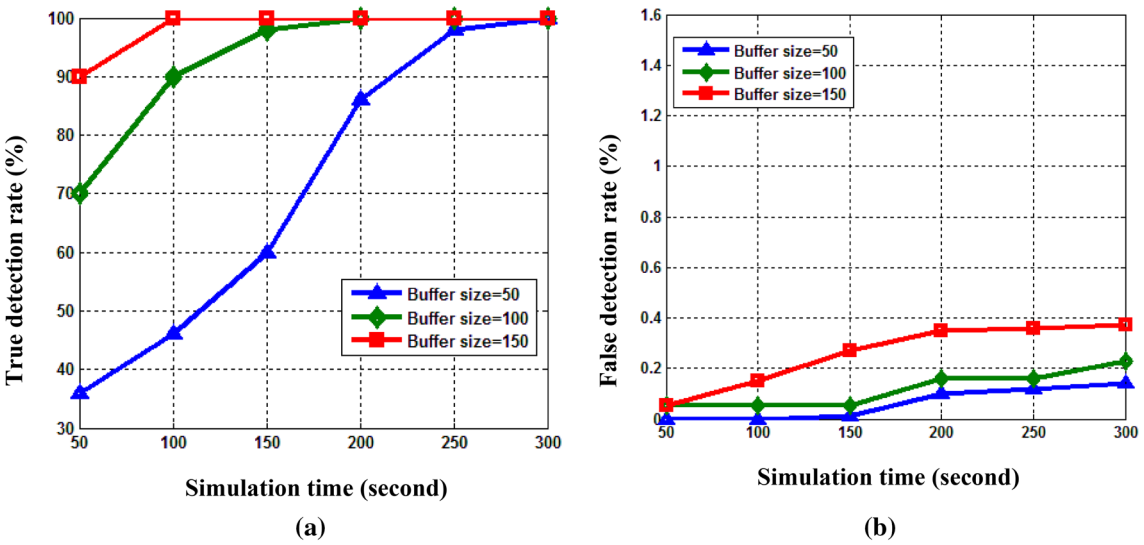


Fig. 6 Effect of the buffer size of watchdog nodes, Buf_{size}, on the true detection rate (**a**) and the false detection rate (**b**) of the proposed algorithm

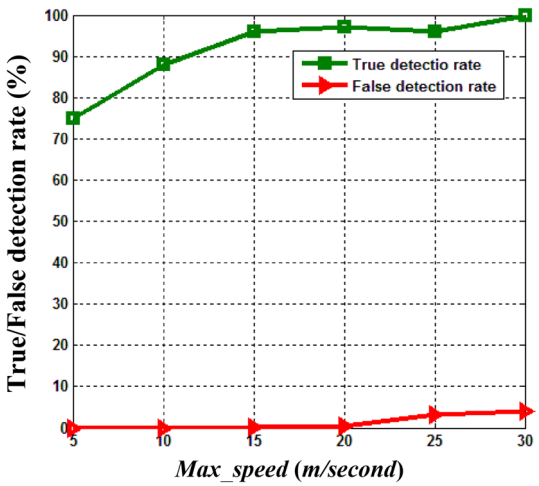


Fig. 7 Effect of the maximum speed of nodes on the true detection rate and false detection rate of the proposed algorithm after 100 s of simulation time

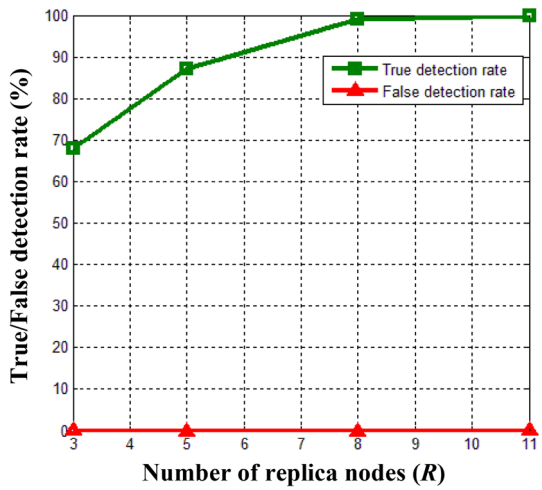


Fig. 8 Effect of the number of replica nodes created from each captured node, R, on the true detection rate and false detection rate of the proposed algorithm after 100 s of simulation time

buffers, the false detection rate increases a little. As mentioned before, by increasing buffer's capacity of each watchdog node, time-location tags are kept in the buffers longer thus false detection rate increases. Indeed, as can be seen in Fig. 6b, changing capacity of watchdog node's buffer, affects false detection rate a little and its rate is less than 0.4% in all cases.

5.3.3 Experiment 3

The purpose of this experiment is to investigate the effect of maximum predefined speed for sensor nodes on the performance of the proposed algorithm. Parameters are set as $n=200$, $R=5$, $M=5$, $w=4$, $Buf_{size}=100$ and the maximum possible speed for nodes varies between 5 m/s and 30 m/s and results are investigated for 100 s of simulation time. Experiment results in terms of both true detection and false detection rate are given in Fig. 7.

The results show that by increasing the maximum possible speed of nodes, both the false detection rate and true detection rate increase. It is clear that by increasing *Max-speed*, watchdog nodes meet each other faster and replica nodes are detected sooner. Additionally, as mentioned in Sect. 4, *Max-speed* is one of the parameters which affects the false detection rate. The results show that the false detection rate for $Max\_speed=5$ m/s

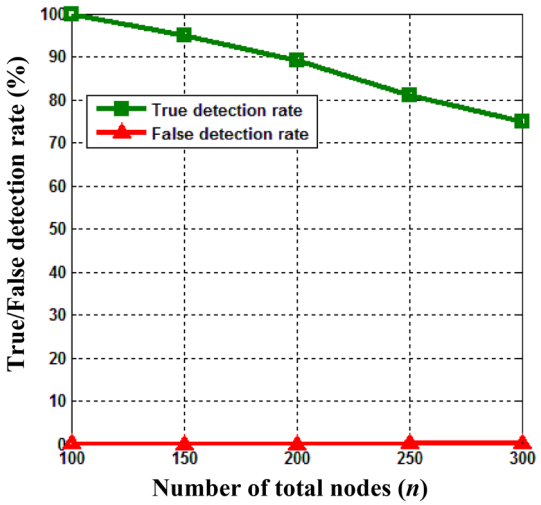


Fig. 9 Effect of the total number of nodes in the network, n, on the false detection and true detection rates of the algorithm after 100 s of simulation time

Table 2 Comparing the performance of the proposed algorithm and other existing algorithms in terms of True detection rate

Algorithm	Parameters	True detection rate (%)
LSM [5]	# line segment = 6	89
RED [7]	–	96
B-MEM [13]	# line segment = 6	86
BC-MEM [13]	# line segment = 5	93
C-MEM [13]	–	95
CC-MEM [13]	–	99
Proposed Alg.	$Buf_{size} = 250, w = 10$ after 50 s of simulation	34
Proposed Alg.	$Buf_{size} = 250, w = 10$ after 100 s of simulation	57
Proposed Alg.	$Buf_{size} = 250, w = 10$ after 150 s of simulation	85
Proposed Alg.	$Buf_{size} = 250, w = 10$ after 200 s of simulation	100

and $Max\text{-}speed = 10$ m/s is 0% and 0.2%, and for $Max\text{-}speed = 30$ m/s is 4%. It should be noted that having sensor nodes with the ability to move with speed of 30 m/s is far from reality.

5.3.4 Experiment 4

The purpose of this experiment is to investigate the effect of the number of replica nodes created from each node captured by the adversary, R, on the performance of the proposed algorithm. In this experiment, parameters are set as $n = 200$, $M = 5$, $w = 4$, $Buf_{size} = 100$, $Max\_speed = 10$ m/s and the number of replica nodes created from each captured node varies from 3 to 11 and the results are evaluated for 100 s of simulation time. Figure 8 shows the results of this experiment in terms of both true detection and false detection rates.

The results show that changing parameter R does not affect the false detection rate of the proposed algorithm, because changing this parameter does not change occurrence probability of a state which results in an error (As mention in Sect. 5.2). On the contrary, changing parameter R affects the true detection rate of the proposed algorithm significantly. For example, if the adversary only creates $R = 3$ copies of each captured node, the true detection rate of the proposed algorithm after 100 s simulation would be 68%. While if it creates $R = 5$, $R = 8$, and $R = 11$ copies of each captured node in the network, this measure would be 87%, 99%, and 100%, respectively. This is because as the number of copies of a specific captured node, u, increases, and node with identity u would be in more regions of the network simultaneously where time-location tag of the node with identity u would be in buffer of watchdog nodes most of the times, thus watchdog nodes detect u being a replica node faster.

5.3.5 Experiment 5

The purpose of this experiment is to investigate the effect of the total number of nodes in the network, n, on the performance of the proposed algorithm. In this experiment, parameters are set as $R = 5$, $M = 5$, $w = 4$, $Buf_{size} = 100$, $Max\_speed = 10$ m/s and the total number of nodes in this network varies from 100 to 300 and the results are investigated for 100 s of the simulation time. Figure 9 shows the results of this experiment in terms of both false detection and true detection rates.

The results show that the false detection rate of the proposed algorithm for different values of n is almost constant and less than 0.1%. But this parameter affects the true detection rate of the proposed algorithm significantly. For example, when $n = 100$, the true detection rate is 100% but for $n = 300$, this rate is 75%, because by increasing the number of nodes in the network, network density is increased and this increases the number of nodes' neighbors. Since watchdog nodes record time-location tag of all neighbors in their buffer after reaching a new location in the network, these tags are not kept for a long time and they are rewritten with new tags very soon. Therefore, the probability of detecting replica nodes decreases. It should be noted when network density is high, the true detection rate of the algorithm can be kept at a favorable level by increasing the monitoring rounds, the buffer size of watchdog nodes, or the number of watchdog nodes.

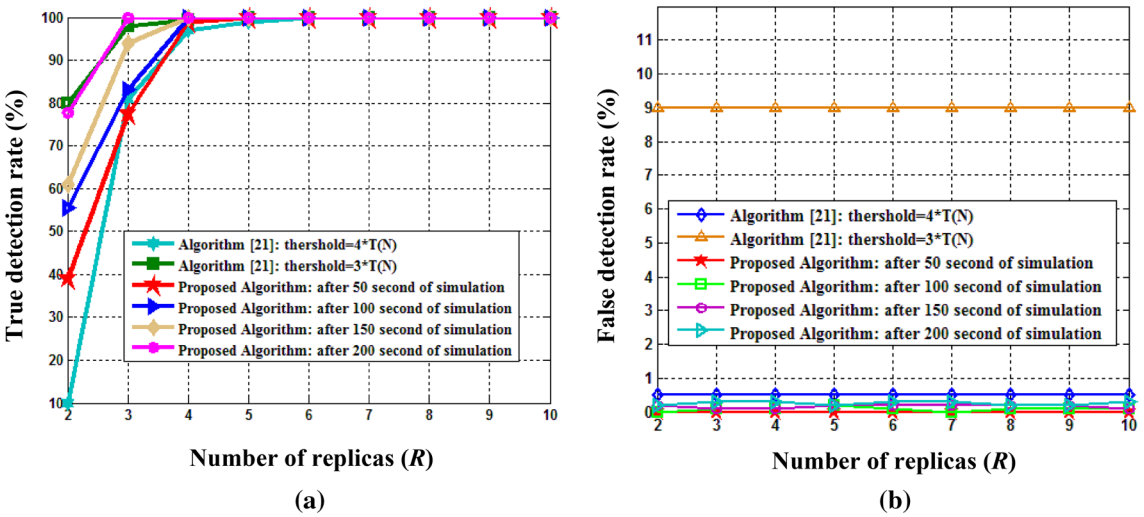


Fig. 10 Comparing the performance of the proposed algorithm and algorithm [21] in terms of the true detection rate (**a**), and the false detection rate (**b**)

5.3.6 Experiment 6

The purpose of this experiment is to compare the performance of the proposed algorithm with some other algorithms (for static networks) in terms of the true detection rate. In this experiment $Max\_speed = 10\,\text{m/s}$ and the total number of nodes is considered to be $n = 1000$. Additionally, the most difficult case for establishing a replica node attack, $M = 1$ and $R = 2$ are also considered. The radio range of nodes is also set such that each node has almost $d = 20$ neighbors. While running the proposed algorithm, nodes are mobile and their maximum speed is considered 20 m/s. But while running other algorithms, nodes are considered to be fixed, because these algorithms are specifically designed for static sensor networks. Table 2 shows a list of evaluated algorithms along with their parameters and the obtained results. As can be seen in the results, LSM, B-MEM, BC-MEM, C-MEM, and CC-MEM algorithms are not able to detect replica nodes 100%, but in the proposed algorithm, as simulation time increases, true detection rate becomes 100%. Indeed, the false detection rate of the static specific algorithms (depending on nature of the algorithm) is usually 0% while for algorithms which are designed for mobile sensor networks, this would be greater than 0%. Thus, in this experiment, the performance of the proposed algorithm is compared with other algorithms only in terms of true detection rate.

5.3.7 Experiment 7

The purpose of this experiment is to compare the performance of the proposed algorithm with the algorithm proposed in [21] (for mobile networks) in terms of both detection and false detection rates. In this experiment, the total number of nodes is $n = 250$ and the radio range of nodes is selected such that each node has almost $d = 12$ neighbors. In this experiment, it is assumed that the adversary has captured $M = 1$ legal node and has created R copies (2 to 10) of that node and injected that to the network. In this experiment, the algorithm of [21] is set with parameters $V_{\min} = 1$, $V_{\max} = 3$, $t_{pause} = 20$ and parameters of the proposed algorithm are set as $w = 6$, $Buf_{Size} = 100$, $Max\_speed = 3\,\text{m/s}$. Figure. 10 shows the results of this experiment in terms of both true detection and false detection rates.

As can be seen in Fig. 10a, if in the algorithm [21], threshold $= 3 * T(N)$, true detection rate would be high and equal to the proposed algorithm (for 200 s of simulation time). But as can be seen in Fig. 10b, the false detection rate of the algorithm [21] is very high, approximately 9% while the false detection rate of the proposed algorithm is less than 1%. Favorable configuration for algorithm [21] is threshold $= 4 * T(N)$ which results in less than 1% false detection rate. Considering such configuration of the algorithm [21], as can be seen in Fig. 10, the proposed algorithm outperforms.

6 Conclusion

In this paper, a novel algorithm using watchdog nodes to detect replica nodes in mobile sensor networks is presented. The main idea of the proposed algorithm is adopted from the fact that replica nodes in a mobile network move faster than the predefined speed because they exist in multiple regions of the network simultaneously. In the proposed algorithm,

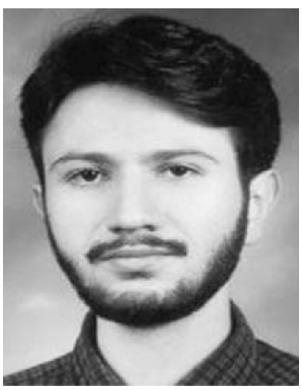
there are a number of watchdog nodes which can detect replica nodes by recording time-location tags for other nodes. The proposed algorithm is implemented by J-SIM simulator and false detection and true detection rates are evaluated through several experiments. Experiment results show that the proposed algorithm performs well in detecting replica nodes.

References

1. M. Jamshidi, E. Zangeneh, M. Esnaashari, A. M. Darwesh and 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 Communications*, Vol. 105, No. 1, pp. 145–173, 2019.
2. M. R. Alagheband and M. R. Aref, Dynamic and secure key management model for hierarchical heterogeneous sensor networks, *IET Information Security*, Vol. 6, No. 4, pp. 271–280, 2012.
3. M. Jamshidi, M. Ranjbari, M. Esnaashari, A. M. Darwesh and M. R. Meybodi, A new algorithm to defend against sybil attack in static wireless sensor networks using mobile observer sensor nodes, *Adhoc & Sensor Wireless Networks*, Vol. 43, pp. 213–238, 2019.
4. C. Karlof and D. Wagner, Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures, *Ad hoc Networks*, Vol. 1, pp. 299–302, 2003.
5. B. Parno, A. Perrig, V. D. Gligor, Distributed detection of node replication attacks in sensor networks. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2005.
6. M. Conti, R. D. Pietro, L. V. Mancini, A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks. In *Proceedings of the ACM MobiHoc*, 2007.
7. M. Conti, R. D. Pietro, L. V. Mancini, A. Mei, Distributed detection of clone attacks in wireless sensor networks. In *Proceedings of the IEEE Transactions on Depedable and Secure Computing*, 2010.
8. H. Choi, S. Zhu, T. F. Porta, SET. detecting node clones in sensor networks. In *Proceedings of the SecureComm*, 2007. pp. 341–350.
9. B. Zhu, V. G. K. Addada, S. Setia, S. Jajodia, S. Roy, Efficient distributed detection of node replication attacks in sensor networks. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2007.
10. C. Kim, C. Park, J. Hur, H. Lee and H. Yoon, A distributed deterministic and resilient replication attack detection protocol in wireless sensor networks, *Communications in Computer and Information Science*, Vol. 56, pp. 405–412, 2009.
11. Y. Zeng, J. Cao, S. Zhang, S. Guo and L. Xie, Random-walk based approach to detect clone attacks in wireless sensor networks, *IEEE Journal on Selected Areas in Communications*, Vol. 28, No. 5, pp. 677–691, 2010.
12. C. M. Yu, C. S. Lu, S. Y. Kuo, CSI: compressed sensing-based clone identification in sensor networks. In *Proceedings of the 8th IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing*, 2012.
13. M. Zhang et al., Memory efficient protocols for detecting node replication attacks in wireless sensor networks. In *Proceedings of the 17th annual IEEE International Conference on Network Protocols, Princeton, NJ, USA*, 2009.
14. M. Jamshidi, A. A. Shaltooki, Z. D. Zadeh and A. M. Darwesh, A dynamic ID assignment mechanism to defend against node replication attack in static wireless sensor networks, *JOIV International Journal on Informatics Visualization*, Vol. 3, No. 1, pp. 13–17, 2019.
15. S. Smys and J. S. Raj, A self-organized structure for mobility management in wireless networks, *Computers & Electrical Engineering*, Vol. 48, pp. 153–163, 2015.
16. Z. Pala, K. Bicakci and M. Turk, Effects of node mobility on energy balancing in wireless networks, *Computers & Electrical Engineering*, Vol. 41, pp. 314–324, 2015.
17. C. M. Yu, C. S. Lu, S. Y. Kuo, Mobile sensor network resilient against node replication attacks. In *Proceedings of the IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2008.
18. J. W. Ho, M. Wright, S. Das, Fast detection of replica node attacks in mobile sensor networks using sequential analysis. In *Proceedings of the IEEE INFOCOM*, 2009. pp. 1773–1781.
19. J. W. Ho, M. Wright and S. Das, Fast detection of mobile replica node attacks in wireless sensor networks using sequential hypothesis testing, *IEEE Transactions on Mobile Computing*, Vol. 10, No. 6, pp. 767–782, 2011.
20. D. Unnikrishnan et al., Detecting mobile replica node attacks in wireless sensor networks using sequential probability ratio test. In *Proceedings of the 13th International Conference on Distributed Computing and Networking (ICDCN), Hong Kong, China*, 2012.
21. X. M. Deng and Y. Xiong, A new protocol for the detection of node replication attacks in mobile wireless sensor networks, *Journal of Computer Science and Technology*, Vol. 26, No. 4, pp. 732–743, 2011.
22. K. Xing and X. Cheng, From time domain to space domain: detecting replica attacks in Mobile Ad Hoc Networks. In *Proceedings of the IEEE INFOCOM*, 2010.
23. C. Zhou and Z. Wang, An two dimension detection to node replication attacks in mobile sensor networks. In *Anti-counterfeiting, Security, and Identification (ASID), 10th IEEE International Conference, 2016*, pp. 63–69.
24. T. Dimitriou, E. A. Alrashed, M. H. Karaata and A. Hamdan, Imposter detection for replication attacks in mobile sensor networks, *Computer Networks*, Vol. 108, pp. 210–222, 2016.
25. M. Conti, R. Di Pietro and A. Spognardi, Clone wars: distributed detection of clone attacks in mobile WSNs, *Journal of Computer and System Sciences*, Vol. 80, No. 3, pp. 654–669, 2014.
26. V. Manickavasagam and J. Padmanabhan, A mobility optimized SPRT based distributed security solution for replica node detection in mobile sensor networks, *Ad Hoc Networks*, Vol. 37, pp. 140–152, 2016.
27. M. Jamshidi, E. Zangeneh, M. Esnaashari and M. R. Meybodi, A lightweight algorithm for detecting mobile Sybil nodes in mobile wireless sensor networks, *Computers & Electrical Engineering*, Vol. 64, pp. 220–232, 2017.
28. M. Jamshidi, A. M. Darwesh, A. Lorenc, M. Ranjbari and M. R. Meybodi, A Precise Algorithm for Detecting Malicious Sybil Nodes in Mobile Wireless Sensor Networks, *IEIE Transactions on Smart Processing & Computing*, Vol. 7, No. 6, pp. 457–466, 2018.
29. B. H. Bloom, Space/time trade-offs in hash coding with allowable errors, *ACM Communication*, Vol. 13, No. 7, pp. 422–426, 1970.
30. M. L. Sichitiu, *Mobility models for ad hoc networks*, Springer, LondonIn Guide to Wireless Ad Hoc Networks, 2009. pp. 237–254.
31. E. Shi and A. Perrig, Designing secure sensor networks, *IEEE Wireless Communications*, Vol. 11, pp. 38–43, 2004.
32. C. Tumrongwittayapak and R. Varakulsiripunth, Detecting sinkhole attacks in wireless sensor networks, In *Proceedings of the ICROS-SICE International Joint Conference, Fukuoka International Congress Center, Japan*, 2009.

33. C. Piro, C. Shields, B. N. Levine, Detecting the sybil attack in Mobile Ad hoc Networks. In *Proceedings of the Securecomm and Workshops*, 2006. pp. 1–11.
34. G. S. Sara and D. Sridharan, Routing in mobile wireless sensor network: a survey, *Telecommunication Systems*, Vol. 57, No. 1, pp. 51–79, 2014.
35. M. Jamshidi, H. Bazargan, A. A. Shaltooki and A. M. Darwesh, A hybrid key pre-distribution scheme for securing communications in wireless sensor networks, *JOIV International Journal on Informatics Visualization*, Vol. 3, No. 1, pp. 41–46, 2019.
36. L. Harn and C. F. Hsu, Predistribution scheme for establishing group keys in wireless sensor networks, *IEEE Sensors Journal*, Vol. 15, No. 9, pp. 5103–5108, 2015.
37. A. Sobeih, J. C. Hou, L. C. Kung, et al., J-Sim: a simulation and emulation environment for wireless sensor networks, *IEEE Wireless Communications*, Vol. 13, No. 4, pp. 104–119, 2006.

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, data mining, and recommender systems.



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.

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

M. R. 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.