Arab J Sci Eng (2018) 43:7229–7251
https://doi.org/10.1007/s13369-017-3052-7

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Guard against trust management vulnerabilities in Wireless Sensor Network

Rashmi Ranjan Sahoo[1] · **Sudhabindu Ray**[2] · **Souvik Sarkar**[3] · **Sourav Kumar Bhoi**[1]

Received: 10 April 2017 / Accepted: 19 December 2017 / Published online: 11 January 2018

Abstract Trust management system in Wireless Sensor Network (WSN) is rudimentary to identify malicious, selfish and compromised nodes. Many proposed trust management systems provide security to WSN from various internal attacks. However, like other security systems, trust management schemes are also vulnerable to attacks. In this paper, we have proposed a lightweight trust management scheme based on penalty and reward policy. We have investigated three different types of attacks against the trust management scheme and provide a defense mechanism for those attacks. The proposed mechanism is based on dynamic time sliding window for calculating the trust value of the nodes and finding out the behavior of the nodes. Theory and simulation show that the proposed GATE method has a high detection rate of the malicious nodes and it demands fewer resources than some of the recently proposed trust management schemes.

Keywords Trust management · Vulnerabilities · Attacks · Trust regain · Dynamic sliding window

1 Introduction

The brisk advances in the collaborative and open environment have increased attention on security issues. The open and collaborative environment makes the Wireless Sensor Network (WSN) easily vulnerable to a variety of internal attacks such as eavesdropping, node compromising and physical disruption [1]. Unfortunately, conventional security mechanism like authentication and cryptography cannot adequately guard against internal attacks [2]. Trust and reputation system has been considered as the principal tool to defend against various insider attacks that happen in WSN [2,3]. The concept of trust has been derived from social science, which helps to improve collaboration between the nodes of WSN by predicting the future behaviors of nodes based on their past interactions. Trust value of a node is directly proportional to the positive interactions between nodes, i.e., trust value of a node increases with increase in positive interaction and vice versa. However, like other security mechanisms, trust management system in WSN is also vulnerable to various attacks [4,5]. Because the primary intention of a malicious node is to remain undetected by performing various kinds of attacks like ON–OFF attack, bad mouthing attack and ballot stuffing attack [4]. Moreover, malicious nodes can deliberately and persistently uphold fewer bad behaviors compared to benevolent behavior so that they cannot be detected easily and can damage the network gradually. In the ON–OFF attack, malicious node behaves alternatively as good and bad. In the case of bad mouthing attack, the primary intention of the malicious node is to provide the malicious recommendation, which degrades the trustworthiness of node. In ballot stuffing attack, the malicious node intends to provide the malicious recommendation, which increases the trustworthiness of node. Among these attacks, the ON–OFF attack is manifested in data forwarding plane. However, bad mouthing attack and ballot stuffing attack are manifested in trust evaluation plane [6,7]. Furthermore, a WSN consists of lots of resource constraint tiny sensors and unreliable radio for communication; it may suffer from unintended transitory errors. Because of that, unintended transitory error node is considered as malicious and debarred from the network by trust management

✉ Rashmi Ranjan Sahoo
rashmiranjan.cse@pmec.ac.in

[1] Department of Computer Science and Engineering, Parala Maharaja Engineering College, Berhampur, Odisha, India

[2] Department of Electronics and Tele-Communication Engineering, Jadavpur University, Kolkata, India

[3] Delloit, India

system. Even after that node returns to its normal behavior, that node may not be used again in the network, which is considered as wastage of system resources [8,9]. To avoid that wastage of system resource, we have used a trust regain scheme. A trust regain scheme provides additional opportunities to those transitory erroneous nodes to regain their trust value by performing some benevolent activities over some period. Although several types of research have been proposed, the defense of trust management scheme has received a very few attention from researchers. Thus, by considering the vulnerability of trust management scheme to various attacks and the resource constraint nature of sensor nodes in WSN, in this paper, we propose a defense mechanism, which can provide shielding to trust management scheme in WSN from attacks mentioned above.

The rest of this article is structured as follows: Section 2 presents our major contribution to this paper. Section 3 discusses related works. The system model used in this article is provided in Sect. 4. Proposed direct trust management scheme and its defense from ON–OFF attacks are shown in Sect. 5. Section 6 presents the indirect trust computation approach. In Sect. 7, theoretical analysis of proposed direct trust and indirect trust computation approach and their performance are presented. Simulation-based performance analysis is discussed in Sect. 8. Section 9 concludes the paper.

2 Our Contributions

The major contributions of this paper are presented as follows.

(i) Periodicity of misbehavior
We have introduced a new factor for trust estimation called as the periodicity of misbehavior, which enables us to mitigate the effect of ON–OFF attackers. Further, the periodicity of misbehavior is also capable of detecting the persistent malicious nodes. The periodicity of misbehavior shows how frequently a node misbehaves in a time window. The consideration of periodicity of misbehavior is more vital than the good behavior of node because the misbehaviors are dangerous.

(ii) Trust regain
Due to unreliable link and open nature of WSN, sometimes nodes of the network may suffer from transitory errors. That error makes the node as transient malicious. Adopted trust regain method enables the trust management system to detect those transient malicious nodes by differentiating from the persistent malicious node and make them as benevolent nodes for further consideration in the network.

(iii) Dynamic sliding time window
Unlike static sliding time window used in trust evaluation of the nodes, we have used the concept of the dynamic sliding time window to detect the low proportional malicious behavior nodes. Because whenever the proportion of malicious behavior is very less, it is quite difficult to detect those persistent malicious nodes.

3 Related Research

Although many research has been carried out on trust model in recent past, however, most of the schemes are lacking in various aspects of trust model in WSN. In this, we have mentioned some of the issues present in recently proposed schemes.

Research work in trust management for WSNs usually based on either direct observation or indirect recommendation or combination of both. In [10], authors proposed a robust trust establishment scheme based on direct observation of nodes behavior only. This scheme captures the node behavior and stores the observation value in a fixed size sliding time window. This scheme lacks in providing defense against insider attacks like bad mouthing and ballot stuffing attack.

In 2013, Li et al. [11] proposed a Lightweight and Dependable Trust System for clustered Wireless Sensor Networks (LDTS), which uses both observation and feedback to calculate the trust. The proposed direct trust computation method uses the strict penalty policy for each bad behavior of the node. This approach uses a self-adaptive weighted method for trust aggregation at cluster head level (CH). However, this scheme lacks in distinguishing between the persistent malicious and transient malicious nodes. Further, in this scheme, the indirect trust is calculated based on the number of positive or negative feedback instead of the actual value of feedbacks.

Shaik et al. [12] proposed a Group-based Trust Management Scheme (GTMS), which is based on both time-based past interaction observation and peer recommendation. This scheme also uses the fixed sliding window to capture the interaction values. This scheme does not provide any mechanism to detect transitory malicious node and filtering of dishonest recommendations.

In [13], Ishmanov et al. proposed a secure trust establishment scheme, which also combines both observation and recommendation to evaluate the trust of a node. Further, this scheme uses a modified M-estimator method to aggregate the recommendations securely. This scheme provides the defense against ON–OFF and bad mouthing attack. The scheme of Ishmanov et al. also fails to distinguish between transient and persistent malicious nodes.

Due to different temporary errors, a node may consider as an untrustworthy node, so a trust regain scheme is essential to allow such node for recovering its trust value. The trust regain method is also known as trust redemption method,

which is broadly categorized into behavior and time-based redemption method [8]. Chae et al. [8] used both behavior and time-based trust redemption scheme. It also uses both good and bad behavior windows to detect ON–OFF attack node.

Many types of research already have been carried to deal with dishonest recommendations. Most of the existing research uses three different approaches to filtering out dishonest recommendations [14]. These are personal experience based [15], majority opinion based [14] and service reputation based [14].

4 System Model

The system model consists of network-related assumptions and node behavioral assumptions.

4.1 Network-Related Assumptions

The proposed model (GATE) is based on the following network model and assumptions.

(i) All sensor nodes in the network are homogeneous with computation capabilities, transmission power, communication range (r) and initial energy.
(ii) All the sensors nodes are static in nature, and all the nodes in the network know their respective position, i.e., x and y coordinates.
(iii) A sensor node can compute the approximate distance from another node by the help of Received Signal Strength Indicator (RSSI).
(iv) A node j is said to be the neighbor of node i if node j falls in the transmission range of node i, i.e., if the Euclidean distance between nodes i and j is less than or equal to communication range (r).
(v) Each sensor node can capture the activities of their neighbor nodes by overhearing the transmission through promiscuous mode.
(vi) Each sensor node has a unique identification number and authentication method to defend the proposed trust model against the Sybil attack.

4.2 Nodes Behavioral Assumptions

The proposed GATE model is based on the behavior of a node in the network. Here, we have categorized the nodes of the network into three different categories, i.e., benevolent/legitimate nodes, persistent malicious nodes and transient malicious nodes. A benevolent node always behaves well; sometimes, it might misbehave transiently because of various factors like channel errors, computation errors or sensing errors. That is the reason why we have assumed that the behavior of benevolent nodes is similar to the behavior of transient malicious nodes or ON–OFF attacking nodes. However, in an ON–OFF attack, the attacking behavior pattern of a node is random, i.e., based on the rate of attack, the pattern of ON and OFF behavior gets changed.

A malicious node exhibits bad behavior persistently by dropping packets, modifying the integrity of data packets, showing selfish behavior to conserve the energy, providing false trust value about benevolent nodes to perform bad and good mouthing attack about well-behaving nodes. However, the periodicity of misbehavior can be either significant or insignificant. Whenever the periodicity of misbehavior is significant (i.e., the proportion of malicious behavior is more), the detection of the malicious node is easier. However, when the malicious nodes exhibit insignificant and persistent misbehavior, it is more challenging to detect them.

5 Proposed Trust Evaluation Method

Like other trust model proposed by various authors, in this section we have also proposed a trust model to compute the direct trust and indirect trust. The proposed direct trust computation scheme protects the trust evaluation model from ON–OFF attack, whereas indirect trust computation scheme provides a guard against bad mouthing and ballot stuffing attacks. Usually, nodes in WSN compute the trust value of their neighbors by observing the weight of transactions. Trust value defines the level of confidence of a node n_i on neighbor node n_j based on the performance of assigned task [16–25]. Further, the trust value of a node is usually expressed as a numerical value. In particular, in WSN trust model deciding the boundary of this trust value is more pivotal because of the resource constraint nature of nodes. In this paper, we have set the sphere of trust value as a positive integer between [0, 10] with an initial trust value of each node is 5, inspired by [21]. Authors Bao et al. [19] and Momani et al. [18] have chosen the sphere of trust value as real numbers (between 0 and 1). In [20], authors have set the trust value as unsigned integer numbers (between 0 and 100). Choosing the trust value as real or unsigned integer increases the communication overhead as well as memory overhead. However, our consideration of positive integer between 0 and 10 has the following advantages.

(i) **Less memory overhead**
An integer representation of trust value between 0 and 100 consumes 1 Byte of memory, and a real-valued representation of trust between 0 and 1 uses 4 Bytes. However, the representation of trust domain between 0 and 10 consumes only one nibble (0.5 Bytes). Therefore, there is a substantial gain of 50 and 87.5% of storage space as compared to integer [0, 100] and real [0, 1], respectively.

(ii) **Less communication overhead**
As the fewer number of bits of the trust value are transmitted from one node to another, the communication overhead is less.

(iii) **Less energy consumption**
As there is involvement of a smaller number of bits of the trust value in transmission, the energy consumption of a sensor node is less.

To keep track of these observations and to compute the trust value of nodes, we have used a dynamic timing sliding window. In [10,11] and [12], authors have used the fixed-sized sliding window to compute the trust value of nodes. In [10,11] and [12], the authors have set the window size to the initialization state of the system, and it remains fixed till the completion of the entire process of trust calculation, but in our case after each time unit the window size either may prolong or reduced based on the behavior of the node. The dynamic timing sliding window has the following advantages over the fixed-sized sliding window.

(i) **Better utilization of system resources**
The system resources have better utilization as compared to fixed size sliding window because in fixed size sliding window if the size is very large, then it is a wastage of system resources.

(ii) **Detection of malicious nodes is effective**
In comparison with the fixed-sized sliding window, the detection of malicious behavior of the nodes is more effective in dynamic timing sliding window because if fixed size is small, then detection of misbehavior by the malicious nodes or benevolent nodes is not effective.

The dynamic timing sliding window consists of a k number of time units, in which each time unit t_k contains the number of successful interactions (good behavior) and unsuccessful interactions (misbehavior) of node i with node j. An interaction between two nodes i and j is said to be successful if the data packets or control packets sent by the node i, received by the node j in the given time to live (TTL); otherwise, the interaction will be called as unsuccessful. Although the dynamic sliding window stores both good and bad behaviors of a node, we are more focused on the bad behavior of a node than good behaviors because bad behaviors of a node are more harmful than good behavior. Figure 1 shows the structure of dynamic timing sliding window. According to Fig. 1, node i stores the observations about node j, in which $S_{i,j}$ and $U_{i,j}$ are the successful and unsuccessful transactions of node j observed by the node i, in time unit t_k. W_L is the sliding window length. The length of the sliding window changes dynamically based on the periodicity of misbehavior of the node (see Sect. 5.2.1). After each time unit, the left boundary of the window moves to the right to forget the previous interaction history and the window length either may prolong or shrinks based on the periodicity of misbehavior. However, the setting of minimum and maximum length of the window depends on system designer, i.e., the boundary of the window is adaptive. The system designer decides the minimum and maximum window size based on the fact that how much damage the system can endure.

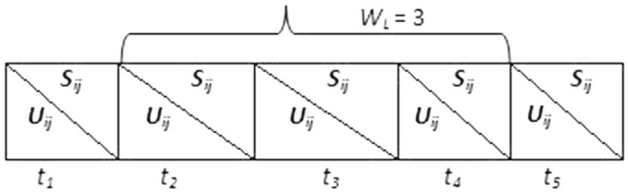

Fig. 1 Structure of the sliding window

5.1 Direct Trust Computation Approach

5.1.1 Direct Trust Calculation

Based on the observed information in a time window, the direct trust value is computed. The direct trust value of node i on node j in a time window (W) is denoted by $\mathrm{DT}_{i,j}(W)$. Since the concept of trust is dynamic, i.e., the trustworthiness either may increase due to the good behavior, or it may decrease due to bad behavior. So in the proposed scheme GATE, we have considered both the reward and penalty policies, i.e., the node that exhibits good behavior is rewarded and the node that shows malicious behavior is penalized. This consideration of both reward and penalty policies in GATE makes it different from other proposed trust computation scheme in WSN. To the best of our knowledge, we are the first to consider both reward and penalty policies for direct trust computation.

$$\mathrm{DT}_{i,j}(W) = \left\lceil \prod_{k=1}^{n} (\mathrm{TF}_k \times \mathrm{DT}_{i,j}(t_k))^{\frac{1}{n}} \right\rceil \tag{1}$$

$$\mathrm{DT}_{i,j}(t_k) = \left\lceil 10 \times \left(\frac{S_{i,j}(t_k)}{S_{i,j}(t_k) + U_{i,j}(t_k)} \right) \left(\frac{S_{i,j}(t_k)}{1 + S_{i,j}(t_k)} \right) \left(\frac{1}{\sqrt{U_{i,j}(t_k)}} \right) \right\rceil \tag{2}$$

where $\mathrm{DT}_{i,j}(t_k)$ is the direct trust value of node i on node j in each time unit t_k and is the direct trust of node i on j within the entire time window W. Initially, the minimum and maximum length of the window (W_L) can be set based on network scenarios. However, with the elapse of time, the prolongation and reduction of W_L depend on the behavior of node (see Sect. 5.2.1). In Eq. (1), n is the number of time unit in time window W and is a trust aging factor, which is

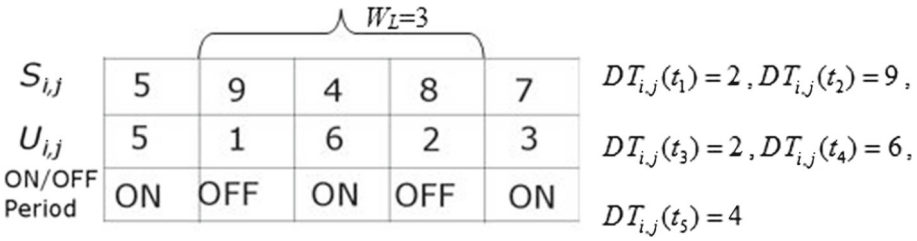

Fig. 2 ON–OFF detection in each time unit

an exponential decrease function and used to ensure that the trust value fades with time. Further, TF_k assigns less weight to older measured trust value so that current performance is given more importance. Therefore, $TF_1 < TF_2 < TF_3 < TF_k$. More specifically, $TF_k = \mu^{n-k}$ where $0 < \mu < 1$. To calculate the direct trust in a time window (W), we use geometric mean of all direct trust computed in each time unit (t_k) instead of simple average (mean). This is because the geometric mean is more suitable for time series data and not sensitive to extreme values of a data set, whereas simple mean is highly sensitive to extreme values of a data set. $\lceil \, . \, \rceil$ represents the nearest integer value, such that $\lceil 4.01 \rceil = 4$. In Eq. (2), $S_{i,j}(t_k)$ is the number of successful interactions of node i with node j during t_k; $U_{i,j}(t_k) \neq 0$ is the number of unsuccessful interactions of node i with node j during time unit (t_k). In a special case, if $S_{i,j}(t_k) \neq 0$ and $U_{i,j}(t_k) = 0$, then we set $\mathrm{DT}_{i,j}(t_k) = 10$. The first term represents the rate of the successful transactions of node i with node j. The second term $\left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right)$ signifies the reward factor. The reward points are given to every successful transaction (good behavior) of node i with j. Further, reward factor also ensures that there is a slow rise in trust value with an increase in good behavior. The third term $\left(\frac{1}{\sqrt{U_{i,j}(t_k)}}\right)$ represents severe punishment factor and also ensures the rapid drop of trust value to zero with an increase in unsuccessful transactions (misbehavior) of node i with j.

5.1.2 Detection of ON–OFF Behavior Using Direct Trust

To detect the ON–OFF attackers, we use an important factor, i.e., the periodicity of misbehavior. The section that follows describes its importance. Since the sliding time window W has several time units t_k, each t_k is set as either an ON or OFF period based on the calculated trust value of node j in that time unit. Hence, attack type of node j in time unit t_k is $A_j(t_k)$, and it is defined as follows:

$$A_j(t_k) = \begin{cases} \text{ON} & \textit{if } DT_{i,j}(t_k) < \lambda \\ \text{OFF} & \textit{otherwise} \end{cases} \tag{3}$$

where λ is the trust threshold of behavior. More specifically, we have set the value as 5, and the reason behind it is discussed in Sect. 7.1. Therefore, if the direct trust value of a node in time unit t_k is less than 5, then the t_k is set as ON period; otherwise, OFF period.

In Fig. 2, node i records the behavior of node j and sets each time unit t_k with either ON or OFF based on Eq. (3). For example, in time unit t_1 the $\mathrm{DT}_{i,j}(t_1) = 2$, which is less than λ (i.e., $\mathrm{DT}_{i,j}(t_1) < 5$), so in time unit t_1 node j shows the ON behavior. Similarly, in t_4 the $\mathrm{DT}_{i,j}(t_4) = 6$, which is greater than λ. Hence, the time unit t_4 of node j is set as OFF period.

5.1.3 Calculation of Periodicity of Misbehavior

The periodicity of misbehavior of a node j ($\mathrm{PM}_j(W)$) is one of the important factors because it says how frequently a node j misbehaves within a time window W. $\mathrm{PM}_j(W)$ is calculated based on the number of ON and OFF periods during a sliding window period W. After setting the entire time unit with either ON or OFF period within the time window W, the periodicity of misbehavior is calculated as follows (Eq. (4)):

$$\mathrm{PM}_j(W) = \frac{\sum \text{ON unit}}{\sum \text{ON unit} + \sum \text{OFF unit}} \tag{4}$$

Based on the value of periodicity of misbehavior of a node within a time W, the status of the node is determined by Eq. (5).

$$S_j(W) = \begin{cases} \text{1. Benevolent Node} & \text{if } 0 \leq \mathrm{PM}_j \leq \theta, \\ \text{2. Persistent Malicious Node} & \text{if } \mathrm{PM}_j = 1, \\ \text{3. Transient Malicious (TM)} & \\ \text{or ON–OFF Attacking Node} & \text{if } \theta < \mathrm{PM}_j < 1. \end{cases} \tag{5}$$

where $S_j(W)$ is the status of the node j in time window W and θ is the threshold value of periodicity of misbehavior. Note that, here the $(0 < \theta < 0.5)$ because, since the entire range of the status of node j is lies between 0 and 1, we have considered the midway for the value of θ. This value of θ is set by the observer node i. Further, a node is considered as benevolent if in each time unit t_k the node exhibits OFF behavior, i.e., $\mathrm{PM}_j \varepsilon (0, \theta)$. On the other hand, the node is considered as persistent malicious if in all time units a node exhibits ON attack, i.e., $\mathrm{PM}_j = 1$. A node is considered as either a transient malicious or ON–OFF attack node if $\mathrm{PM}_j \varepsilon (\theta, 1)$. In the next sec-

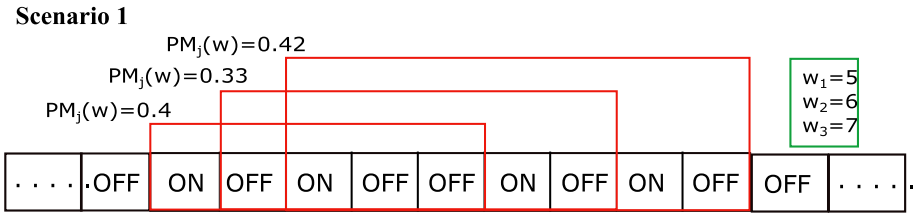

Fig. 3 Size of window (prolonged) and TM/ON–OFF node confirmed as ON–OFF attacking node

tion, we have shown the procedure to differentiate the transient malicious (TM) and ON–OFF attacking node by using trust regain method with dynamic timing sliding window.

5.2 Trust Regain

It is utmost important to find the actual status of a node, i.e., whether the node is a transient malicious or ON–OFF attacker because a node whose status is transient malicious may recover its trust value by exhibiting good behavior over some time to become benevolent. However, a node which is ON–OFF attacker cannot recover its trust. Hence, this proposed trust regain method can help the transient malicious node to recover its trust value over some time and to become benevolent (i.e., to decrease their periodicity of misbehavior). However, sometimes it may so happen that a transient malicious node may not recover its trust value within the chance given. Further, the transient malicious node may convert to persistent malicious within the chance given. On the other hand, an ON–OFF attacking node launches the attack in a preset manner (i.e., its periodicity of misbehavior remains almost constant).

Therefore, the nodes with transient malicious/ON–OFF attacking status are given some more time unit, and their periodicity of misbehavior is calculated. If the updated value of the periodicity of misbehavior in the subsequent time window is:

(i) Remains almost same, as previously, then the node will be conferred as an ON–OFF attacking node.
(ii) Float between θ and 1, then the status of the node will be conferred as transient malicious.
(iii) Decreases and reaches the threshold θ or below θ, then the status of the node will be changed from transient malicious to the benevolent node.
(iv) Increases and reaches to 1, then the status of the node will be changed from transient malicious to the persistent malicious node.

5.2.1 *Update of Periodicity of Misbehavior Using Dynamic Time Sliding Window*

To update the periodicity of misbehavior of a node, we resized the sliding window size into kind of dynamic timing sliding window. The change in the size of the sliding window depends on the following two cases.

(i) When the number of ON period (misbehavior) is less than or equal to the number of OFF period (good behavior) within the time window, the window size is prolonged by one-time unit at a time till it reaches the maximum size of window length, to cover more observed behavior.
(ii) When the number of ON period (misbehavior) is more than the number of OFF period (good behavior) within the time window, the window size is reduced by one at a time till it reaches the minimum size of window length because the smaller size of the sliding window can detect the misbehavior effectively.

Therefore, mathematically size of dynamic timing sliding window is defined as:

$$\text{ON period } \alpha \frac{1}{W_L} \tag{6}$$

According to Eq. (6), the size of sliding window is inversely proportional to ON Attack in that window, i.e., if the number of ON period is more than number of OFF period in a window, then the size of the window (W_L) is reduced, and if the number of OFF period is more than ON period, then the size of the window (W_L) is prolonged. Some sample scenario of dynamic timing sliding window usage, for calculating the updated value of $\text{PM}_j(W)$ and detecting the transient and ON–OFF attacking nodes are present through an example. An illustrative example is shown in four different scenarios in Figs. 3, 4, 5 and 6. In this example, the minimum $W_L = 3$ and maximum $W_L = 7$, and initially, we consider the window size is five and threshold of the periodicity of misbehavior $\theta = 0.3$.

According to Figs. 3, 4, 5 and 6, the size of the dynamic sliding window is changed based the ON attack period. Scenario 1 shows a node with transient malicious/ON–OFF attacking status has given some more chances to decrease its periodicity of misbehavior and to become benevolent, but node could not decrease its periodicity of misbehavior; rather, it maintains the almost same periodicity of misbehavior. So the node finally confirmed as an ON–OFF attacking node (i.e., not a transient malicious node).

Scenario 1, in Fig. 3, node i observes the behavior of node j and node i suspects that node j launches two ON and three OFF attacks, with initial window length $W_1 = 5$ and $\text{PM}_j(W) = 0.4$. As the $\text{PM}_j(W)$ lies in $[\theta, 1]$, node i sets

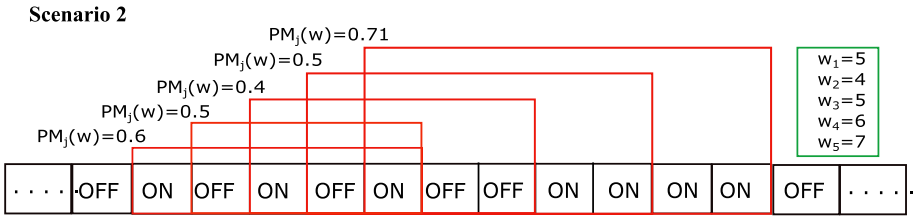

Fig. 4 Size of window (prolonged & reduced) and TM converted to transient malicious

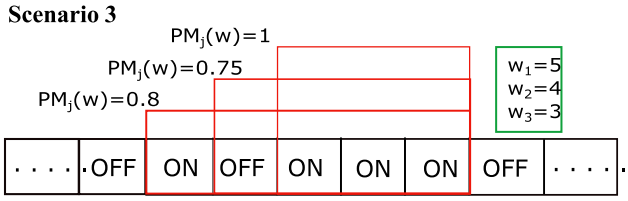

Fig. 5 Size of window (reduced) and TM converted to persistent malicious

the status node j to transient malicious/ON–OFF attacking node. Further, as the first window has less number of ON period than OFF period, node i prolonged the window size by one-time unit, which results in the second window length $W_2 = 6$ with updated $\mathrm{PM}_j(W) = 0.33$. Further, in second window node i found that node j has two ON units and four OFF units, so node prolonged the second window to the third window with $W_3 = 7$ and updated $\mathrm{PM}_j(W) = 0.42$. Since the prolongation of window reaches to its maximum length (i.e., seven), final updated value of periodicity of misbehavior is 0.42 (which is almost similar to the initial periodicity of misbehavior value, i.e., 0.4). Therefore, node i confirms that node j is an ON–OFF attacking node.

According to scenario 2, Fig. 4, the first window has three ON and two OFF units with window length $W_1 = 5$ and $\mathrm{PM}_j(W) = 0.6$. As the $\mathrm{PM}_j(W)$ lies in $[\theta, 1]$, the status of the node j is transient malicious/ON–OFF attacking node. Since in the first window, the number of ON unit is more than the OFF unit, node i reduces the size of the window by one unit, which results in the second window with length $W_2 = 4$. Further, the second window has two ON units and two OFF units with updated $\mathrm{PM}_j(W) = 0.5$. As the second window has an equal number of ON and OFF units, the second window prolonged by 1 unit, which results in the third window with $W_3 = 5$, updated $\mathrm{PM}_j(W) = 0.4$ and two ON units and three OFF units. As the third window has two ON and three OFF units, the third window prolonged by 1 unit, which results in the fourth window with $W_4 = 6$, updated $\mathrm{PM}_j(W) = 0.5$ and three ON units and three OFF units. Further, as the fourth window has an equal number of ON and OFF units, the fourth window prolonged by 1 unit, which results in the fifth window with $W_5 = 7$, updated $\mathrm{PM}_j(W) = 0.71$. Now, the window length reaches its maximum value, i.e., 7 with a final updated value of periodicity of misbehavior which is 0.71 (which lies in $[\theta, 1]$). Therefore, node i confirms that node j is transient malicious.

In scenario 3, a transient malicious node could not regain its trust value in the given a chance and become persistent malicious. In Fig. 5, the first window has four ON and one OFF units with window length $W_1 = 5$ and $\mathrm{PM}_j(W) = 0.8$. As the $\mathrm{PM}_j(W)$ lies in $[\theta, 1]$, the status of the node j is transient malicious/ON–OFF attacking node. Since in the first window, the number of ON unit is more than the OFF unit, node i reduces the size of the window by 1 unit, which results in the second window with length $W_2 = 4$. Further, the second window has three ON units and one OFF unit with updated $\mathrm{PM}_j(W) = 0.75$. Since the second window has more ON unit than OFF unit, the window is again reduced by 1 unit, which results in $W_3 = 3$ and $\mathrm{PM}_j(W) = 1$. Now, the window length reaches its minimum value (i.e., 3) with a final updated value of periodicity of misbehavior which is 1. Therefore, the status of that node j changes from transient malicious to persistent malicious.

In scenario 4, a transient malicious node regains its trust value by showing good behavior with the given a chance and becomes a benevolent one. In Fig. 6, the final window has length $W_5 = 7$ with final updated $\mathrm{PM}_j(W) = 0.28$. As the window reaches its maximum length (i.e., 7) and $\mathrm{PM}_j(W)$ lies $[0, \theta]$, the status of that node j changes from transient malicious to benevolent.

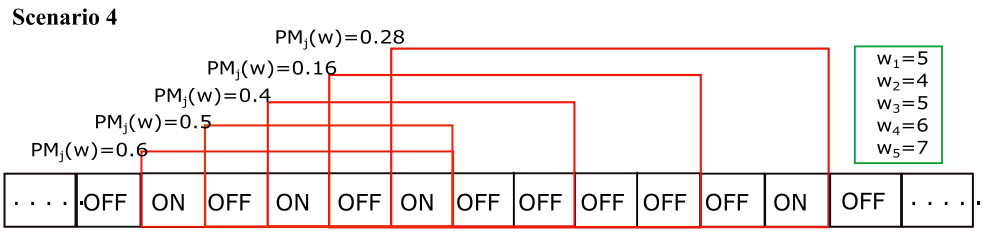

Fig. 6 Size of window (prolonged and reduced) and TM/ON–OFF converted to benevolent

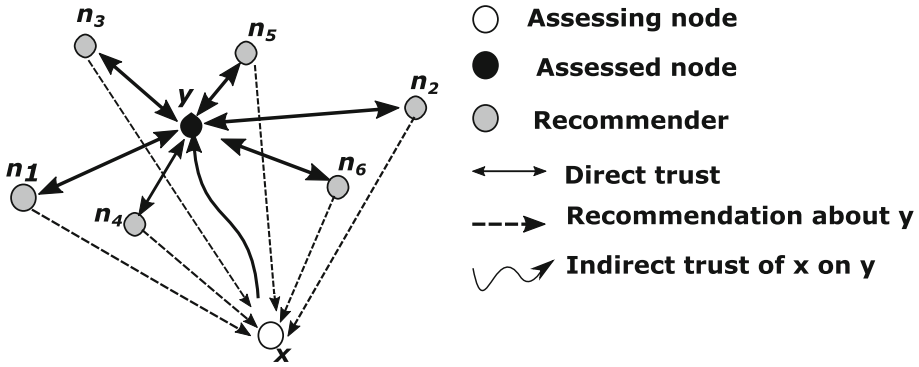

Fig. 7 Indirect trust

Fig. 8 Indirect trust computation with honest recommendations

6 Indirect Trust Computation Approach

Direct trust value of a node is computed completely based on node behavior by observations. However, the indirect trust value is calculated based on recommendations provided by the neighbors. Whenever a node (assessing node) has no personal observation with another node (assessed node) to compute the direct trust value, assessing node seeks the recommendations from the neighbors of the assessed node to calculate the indirect trust value. Indirect trust can provide the confidence to a node to interact with unknown nodes in a network. However, the computation of indirect trust through recommendations is vulnerable to various kinds of attacks like bad mouthing attack and ballot stuffing attack.

In Fig. 7, node x (assessing node) has no personal experience with node y (assessed node). Whenever node x wants to interact with node y, it has to compute the indirect trust value of node y by gathering the recommendations from common neighbors (recommender) of node x and node y, and the recommendations given by the neighbors to node x are the direct trust value on node y. Further, the recommendations provided by each neighbor can be either honest or dishonest. Hence, for calculating the indirect trust, dishonest recommendations are filtered out as outliers. Figure 8 shows the flow of indirect trust calculation with honest recommendations.

In the proposed indirect trust computation scheme, the set of recommendations given by all recommenders for the assessed node is denoted by the set RTS and the set TE represents the set of trust value of assessing node on each of the recommenders. In another word, the set TE denotes the trustworthiness of each recommender. The set RTS $= \mathrm{RT}_1, \mathrm{RT}_2, \mathrm{RT}_3, ...\mathrm{RT}_{n-1}, \mathbf{RT}n$ and set TE $= \mathrm{TE}_1, \mathrm{TE}_2, \mathrm{TE}_3, ...\mathrm{TE}n-1, \mathrm{TE}_n$ are given as input to recommendation filter module, and it segregates the honest recommendations from dishonest. Finally, all the filtered honest recommendations are gathered for indirect trust computation.

6.1 Recommendation Filtering

This section presents the segregation of honest and dishonest recommendations from entire set RTS by using a modified median of median absolute deviation (MADAM) statistical technique, inspired by Huber et al. [26]. To calculate (MADAM) for the given recommendation set (RTS), the absolute deviation from the median for each of the data in RTS is calculated by Eq. (7).

$$|\mathrm{RT}_i - \mathrm{median(RTS)}|\,, \text{for } i = 1, 2, 3...n \tag{7}$$

where RT_i is the recommendation given by ith node about the assessed node. Now, MADAM is determined as Eq. (8)

$$\mathrm{MADAM} = \mathrm{median}\,|\mathrm{RT}_i - \mathrm{median(RTS)}| \tag{8}$$

With the help of MADAM, we calculate the variation factor (VF), which gives the notion of deviation of TE from MADAM. The variation factor is defined as the ratio between MADAM and TE.

$$\mathrm{VF}_i = \frac{\mathrm{MADAM}}{\mathrm{TE}_i} \tag{9}$$

In Eq. (9), VF_i is the variation factor which signifies the dissimilarity of each received recommendation from MADAM and value of $\mathrm{TE}_i \neq 0$ (i.e., if the trust of assessing node is zero on any of the recommender nodes then the recommendation given by that node is ignored). Based on the calculated value of VF_i, the type of recommendation (i.e., honest or dishonest) is determined by Eq. (10).

$$\text{recommendation\_type} = \begin{cases} \text{Dishonest recommender} & \text{if } \mathrm{VF}_i \geq C \\ \text{Honest recommender} & \text{Otherwise} \end{cases} \tag{10}$$

where C is the threshold value of variation factor. The value of threshold C depends on the system designer. A system designer can choose the appropriate value of C and can verify how tolerant the trust management system will be to different attacks. The analytical section shows that which value of threshold C is more suitable for system tolerance. Based on the recommendation type, one recommender either will fall into the category of honest recommendation set (HRS) or dishonest recommender set (DRS). Section 6.2 shows an illustrative example of separating the honest recommendations from dishonest recommendations. Moreover, Eq. (9) signifies that if the trust value of assessing node on any of the recommender is more, then the value of variation factor is less and vice versa. Further, it is noteworthy that the computation of MADAM considers the views of all recommenders (i.e., median of RTS), and on the other hand, calculation of VF considers the personal interaction experience on each recommender (i.e., set TE). The combination of all recommender's views and personal interaction experiences makes the indirect trust computation unbiased. If the calculation VF only depends on views of all recommenders, then there is a chance that whenever most of the recommendations are malicious; the indirect trust computation may negatively influence. Similarly, if the calculation VF only depends on the personal interaction experience of evaluating node, then there is a chance that smart attacker may give dishonest trust value to assessing node after winning its confidence.

6.1.1 Indirect Trust Computation

After filtering out all dishonest recommenders, the indirect trust of assessing node on the assessed node is computed by using the recommendations of honest recommender only. The indirect trust can be calculated by Eq. (11):

$$\mathrm{IT}_{x,y} = \frac{\sum_{i=1}^{m} \sqrt{\left(\mathrm{TE}_{x,\mathrm{HRS}_i} \times \mathrm{avg(HRS)}\right)}}{|m|} \tag{11}$$

where $\mathrm{IT}_{x,y}$ is the indirect trust of node x on node y. $\mathrm{TE}_{x,\mathrm{HRS}_i}$ is the direct trust value of node x on each honest recommender. $|m|$ represents cardinality of honest recommender set (HRS).

6.2 An Illustrative Example

In this section, the proposed recommendation filtering technique is validated by taking an illustrative example for filtering out the all the dishonest recommender. Let us consider a node x (assessing node) which has no prior interaction with node y (assessed node). Now, node x wants to interact with node y. Therefore, node x has to evaluate the trustworthiness of node y. For evaluating the trust worthiness of node y, node x seeks the recommendation from the common neighbors $n_i = 16$.

Let $\mathrm{RTS} = \{\mathrm{RT}_1, \mathrm{RT}_2, \mathrm{RT}_3 ... \mathrm{RT}_{15}, \mathrm{RT}_{16}\}$ be a recommendation set received by node x from the common neighbor nodes $n_1, n_2, n_3 ... n_{16}$ about node y. Further, set $\mathrm{TE} = \{\mathrm{TE}_1, \mathrm{TE}_2, \mathrm{TE}_3 ... \mathrm{TE15}, \mathrm{TE}_{16}\}$ is the direct trust value of node x on each recommender $n_1, n_2, n_3 ... n_{16}$.

Illustration:

Let $\mathrm{RTS} = \{0, 0, 2, 1, 3, 4, 5, 4, 6, 7, 7, 8, 9, 10, 9, 10\}$, is the recommendation set.

$\mathrm{TE} = \{1, 4, 2, 8, 4, 2, 5, 9, 7, 8, 3, 9, 6, 10, 9, 9\}$, is the set of trust value of evaluating node on recommender. After finding the value of MADAM using Eq. (8) (see column 4, Table 1), the value of variation factor (VF) is computed for each recommender node using Eq. (9). In order to identify the dishonest recommender, the value of VF_i is compared with a threshold value C, Eq. (10). If the value of VF_i is greater than or equal to C, then the recommender is considered as dishonest one. Here, we have considered the value of threshold $C = 1$. By comparing the VF_i with C, we found the dishonest recommender set $\mathrm{DRS} = \{n_1, n_3, n_6, n_{11}\}$ and honest recommendation set $\mathrm{HRS} = \{n_2, n_4, n_5, n_7, n_8, n_9, n_{10}, n_{12}, n_{13}, n_{14}, n_{15}, n_{16}\}$. In Table 1 columns 6 and 7, the symbols Y and N signify that the particular recommender falls under DRS or HRS.

7 Theoretical Analysis and Performance Evaluation of GATE

In this section, we analyze and proof that our proposed GATE scheme provides a guard to the trust management system against the malicious node. This analysis section analyzes the robustness of the GATE in two ways. Firstly, we analyze the direct trust evaluation method and prove how the direct trust evaluation could provide a guard to the trust management against the malicious node. Secondly, we analyze the indirect trust evaluation method to show the robustness of proposed trust management system. To ana-

Table 1 An example of finding the trustworthiness of recommender

Recommender node	Recommender set (RTS)	TE	RTi-median (RTS)	Variation factor (VF)	DRS	HRS
n_1	0	1	5.5	3	Y	N
n_2	0	7	5.5	0.428571	N	Y
n_3	2	2	3.5	1.5	Y	N
n_4	1	8	4.5	1.5	Y	N
n_5	3	6	2.5	0.5	N	Y
n_6	4	2	1.5	1.5	Y	N
n_7	5	5	0.5	0.5	N	Y
n_8	4	9	1.5	0.3333	N	Y
n_9	6	7	0.5	0.428571	N	Y
n_{10}	7	8	1.5	0.375	N	Y
n_{11}	7	3	1.5	1	Y	N
n_{12}	8	9	2.5	0.3333	N	Y
n_{13}	9	6	3.5	0.5	N	Y
n_{14}	10	10	4.5	0.3	N	Y
n_{15}	9	9	3.5	0.3333	N	Y
n_{16}	10	9	4.5	0.3333	N	Y

lyze the GATE protocol, we broadly categorized the nodes of WSN into two types of node: benevolent and persistent malicious. Our assumption is that a benevolent node always exhibits OFF behaviors and provides honest recommendations. However, persistent malicious nodes always exhibit ON behaviors and give manipulated recommendations by launching various attacks. Note that, in this section, we do not distinguish between persistent and transient malicious nodes. In Sect. 5.2.1, we have already shown how to differentiate between transient malicious and persistent malicious nodes. Further, in Sect. 8, we capture the behavior of malicious nodes for providing the guard to trust management system against various attacks.

7.1 Analysis of Direct Trust Evaluation Scheme

Definition 1 In the GATE protocol, node j exhibits ON behavior in a time unit t_k, when the measured direct trust value of node i on node j in time unit t_k is less than or equal to a threshold $\lambda = 5$ (i.e., $A_j(t_k)$ is ON when $\mathrm{DT}_{i,j}(t_k) \leq 5$). Further, for ON behavior the number of unsuccessful interaction is more than the successful interaction ($U_{i,j}(t_k) > S_{i,j}(t_k)$).

Definition 2 In the GATE protocol, node j exhibits OFF behavior in a time unit t_k, when the measured direct trust value of node i on node j in time unit t_k is greater than a threshold $\lambda = 5$ (i.e., $A_j(t_k)$ is OFF when $\mathrm{DT}_{i,j}(t_k) > 5$). Since a time window W is consist of many time unit t_k, a node is considered as a benevolent or malicious based on its periodicity of ON or OFF behavior in a time window.

Definition 3 A node j is said to be benevolent if it always shows OFF behavior in a time window W with any other node i and also gives honest recommendations for indirect trust computation.

Definition 4 A node j is said to be persistent malicious if it always shows ON behavior in a time window W with any other node i and also gives dishonest recommendations for indirect trust computation.

Lemma 1 *The proposed direct trust computation method prevents the malicious nodes to deceive the trust management system.*

Proof To prove Lemma 1, we use the method of contradiction. Assume that a malicious node j deceives the trust management system. Then, according to definition 3, the malicious node j always shows OFF behavior in a time window W to be benevolent and remains undetected, and according to definition 2, for malicious node j all the time units t_k of the sliding window W must be OFF, i.e., $\mathrm{DT}_{i,j}(t_k) > 5$. Therefore, we prove that the computed direct trust value of malicious node j in all the time units t_k can never be greater than five, i.e., $\mathrm{DT}_{i,j}(t_k) \leq 5$. To prove the lemma by contradiction, we consider the definition 2, in which $\mathrm{DT}_{i,j}(t_k) > 5$ and $U_{i,j}(t_k) > S_{i,j}(t_k)$ follow three cases:

Case 1 If $S_{i,j}(t_k) > 1$, i.e., node i has some successful interaction with node j in time unit t_k.

Let $p = \frac{U_{i,j}(t_k)}{S_{i,j}(t_k)}$ Given that $U_{i,j}(t_k) > S_{i,j}(t_k)$, we can derive that $p > 1$. So, $U_{i,j}(t_k) + S_{i,j}(t_k) \neq 0$.

According to Eq. (2), the direct trust in a time unit t_k is:

$$\mathrm{DT}_{i,j}(t_k) = \left\lceil 10 \times \left(\frac{S_{i,j}(t_k)}{S_{i,j}(t_k)+U_{i,j}(t_k)}\right)\left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right)\left(\frac{1}{\sqrt{U_{i,j}(t_k)}}\right)\right\rceil$$

$$= \left\lceil 10 \times \left(\frac{1}{1+\frac{U_{i,j}(t_k)}{S_{i,j}(t_k)}}\right) \times \left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right) \times \left(\frac{1}{\sqrt{p \times S_{i,j}(t_k)}}\right)\right\rceil$$

As, $p = \frac{U_{i,j}(t_k)}{S_{i,j}(t_k)}$, $\therefore U_{i,j}(t_k) = p \times S_{i,j}(t_k)$

$$= \left\lceil 10 \times \left(\frac{1}{1+\frac{U_{i,j}(t_k)}{S_{i,j}(t_k)}}\right) \times \left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right) \times \left(\frac{1}{\sqrt{p \times S_{i,j}(t_k)}}\right)\right\rceil$$

$$= \left\lceil \left(\frac{10}{1+p}\right)\left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right) \times \left(\frac{1}{\sqrt{p \times S_{i,j}(t_k)}}\right)\right\rceil$$

$$= \left\lceil \frac{10}{(1+p)\left(1+S_{i,j}(t_k)\right)\left(\frac{\sqrt{p \times S_{i,j}(t_k)}}{S_{i,j}(t_k)}\right)}\right\rceil$$

$$= \left\lceil \frac{10}{(1+p)\left(1+S_{i,j}(t_k)\right)\left(\frac{\sqrt{p}}{\sqrt{S_{i,j}(t_k)}}\right)}\right\rceil$$

$$= \left\lceil \frac{10}{(\sqrt{p})(1+p)\left(\frac{1}{\sqrt{S_{i,j}(t_k)}}+\sqrt{S_{i,j}(t_k)}\right)}\right\rceil$$

As, $p > 1$ and $S_{i,j}(t_k)$

So, the terms of denominator

$(\sqrt{p}) > 1$, $(1+p) \geq 2$ and

$$\left(\frac{1}{\sqrt{S_{i,j}(t_k)}}+\sqrt{S_{i,j}(t_k)}\right) \geq 2$$

$$\mathrm{DT}_{i,j}(t_k) = \left\lceil \frac{10}{(\sqrt{p})(1+p)\left(\frac{1}{\sqrt{S_{i,j}(t_k)}}+\sqrt{S_{i,j}(t_k)}\right)}\right\rceil \geq \left\lceil \frac{10}{4}\right\rceil$$

Since it is given that $\mathrm{DT}_{i,j} > 5$, it yields a contradiction. Thus, it proves our Lemma 1. □

Case 2 If $S_{i,j}(t_k) = 0$ and $U_{i,j}(t_k) > 1$, i.e., node i has no successful interaction with node j, but has some unsuccessful interaction with node j in time unit t_k.

Proof According to Eq. (2), the direct trust in a time unit t_k is:

$$\mathrm{DT}_{i,j}(t_k) = \left\lceil 10 \times \left(\frac{S_{i,j}(t_k)}{S_{i,j}(t_k)+U_{i,j}(t_k)}\right)\left(\frac{S_{i,j}(t_k)}{1+S_{i,j}(t_k)}\right)\left(\frac{1}{\sqrt{U_{i,j}(t_k)}}\right)\right\rceil = 0$$

which signifies that if a node i has no successful interaction with another node j in time unit t_k, then node i exhibits ON behavior only in t_k, so $\mathrm{DT}_{i,j}(t_k)$ is zero, which contradicts our assumption; thus, prove Lemma 1. □

The following case 3 is a special case.

Case 3 if $S_{i,j}(t_k) = 0$ and $U_{i,j}(t_k) = 0$, i.e., there is no interaction between nodes i and j in time unit t_k. As there are no interactions in t_k, there may be a chance that node i may interact with node j in other time unit t_k of sliding window W. If it interacts, then either through case 1 or case 2, its maliciousness can be proved. If it does not interact throughout the sliding window W, then indirect trust is calculated through recommendations.

7.2 Analysis of Indirect Trust Evaluation

Definition 5 In the GATE protocol, an honest recommender n_1 always gives the trust value greater than or equal to five, about a benevolent assessed node y.

Definition 6 In the GATE protocol, an honest recommender n_1 always gives the trust value less than five, about a malicious assessed node y.

Lemma 2 *In the indirect trust computation, proposed GATE is robust against dishonest recommender.*

Proof To prove Lemma 2, the following cases are considered:

Case 4 Honest recommendations about a benevolent assessed node always yield the indirect trust $\mathrm{IT}_{x,y} \geq 5$, where x is the assessing node and y is the assessed node.

Case 5 Honest recommendations about a malicious assessed node always yield the indirect trust $\mathrm{IT}_{x,y} < 5$, where x is the assessing node and y is the assessed node.

Both case 4 and case 5 of Lemma 2 are proved in the worst case, average case and best case.

According to Eq. (11), the indirect trust is:

$$\mathrm{IT}_{x,y} = \frac{\sum_{i=1}^{m}\sqrt{(\mathrm{TE}_{x,\mathrm{HRS}_i} \times \mathrm{avg(HRS)})}}{|m|}$$

Let n be the number of honest recommender, which gives the honest recommendation about benevolent assessed node y, so

$$\mathrm{IT}_{x,y} = \frac{\sqrt{\mathrm{TE}_{x,\mathrm{HRS}_1} \times \mathrm{avg(HRS)}} + \sqrt{\mathrm{TE}_{x,\mathrm{HRS}_2} \times \mathrm{avg(HRS)}} + \cdots + \sqrt{\mathrm{TE}_{x,HRS\mathrm{HRS}_n} \times \mathrm{avg(HRS)}}}{n}$$

Worst case: The recommendation given by all honest recommender is five. So, the term avg(HRS) = 5. Since all the recommenders are honest, the trust of node x on each recommender is 5. Hence, the term $\mathrm{TE}_{x,HRS_i} = 5$.

$\Rightarrow IT_{x,y} = \frac{(\sqrt{(5\times5)}+\sqrt{(5\times5)}+\cdots+\sqrt{(5\times5)})\text{for } n \text{ times}}{n}$
$= \frac{n\times\sqrt{25}}{n} = 5$, which proves the case 4 of Lemma 2 in the worst case.

Average case: The recommendation given by all honest recommender is greater than five, but less than ten. The proof of case 4 in average case is similar to the worst case.

Best case: The recommendation given by all honest recommenders is 10. So, the term avg (HRS) = 10. Since all the recommenders are honest, the trust of node x on each recommender is 10. Hence, the term $\mathrm{TE}_{x,\mathrm{HRS}_i} = 10$.
$\Rightarrow \mathrm{IT}_{x,y} = \frac{(\sqrt{(10\times10)}+\sqrt{(10\times10)}+\cdots+\sqrt{(10\times10)})\text{for } n \text{ times}}{n}$
$= \frac{n\times\sqrt{100}}{n} = 10$, which proves the case 1 in the best case. □

Similarly, case 5 of Lemma 2 is proved through the worst case, average case and best case.

Proof **Worst case:** The recommendation given by all honest recommenders is zero. So, the term avg (HRS) = 0.
$\Rightarrow \mathrm{IT}_{x,y} = 0$

Average case: The recommendation given by all honest recommender is greater than zero, but less than four.
Proof of case 5 in average case is similar to the worst case.

Best case: The recommendation given by all honest recommenders is four. So, the term avg (HRS) = 4. Since all the recommenders are honest, the trust of node x on each recommender is 4. Hence, the term $\mathrm{TE}_{x,\mathrm{HRS}_i} = 4$.
$\Rightarrow \mathrm{IT}_{x,y} = \frac{(\sqrt{(4\times4)}+\sqrt{(4\times4)}+\cdots+\sqrt{(4\times4)})\text{for} n\text{times}}{n}$
$= \frac{n\times\sqrt{16}}{n} = 4$, which proves the case 2 is the best case.
As all two cases are proved in the worst case, average case and best case, Lemma 2 is proved. □

7.3 Overhead Analysis

This section presents the memory overhead and communication overhead for the entire network. Both the overhead are analyzed, simulated and compared with other two schemes LDTS [11] and GTMS [12] under the following network scenario.

Overhead is analyzed under varying number of nodes $N = \{100, 150, 200, 250\}$. It is also assumed that the entire network is divided into some clusters (M)=10 for LDTS and GTMS (i.e., number of cluster head (CH) is also 10). Therefore, the average number of node in a cluster $\delta = \{10, 15, 20, 25\}$, respectively. Further, the number of neighbor node of any node is $N_a = \{5, 10, 15, 20\}$, respectively, which is also considered. Let the average number of hops $h = 3$ be in between any sensor node and base station (BS). We have assumed that every node in the network wants to communicate with BS in h hops.

7.3.1 Communication Overhead

In this section, we evaluate the communication overhead of our proposed GATE mechanism under a worst case scenario, in which a node x wants to communicate with all of its neighbor nodes. The communication overhead is influenced by the total number of messages exchanged by the sensor node with its neighbor to execute the GATE protocol.

When a node x wants to communicate with its neighbor node y (which is an unknown for node x), node x seeks recommendations from the neighbors of node y. Let N_a be the number of neighbors of node y. Therefore, in the worst case, node x will send maximum N_a number of recommendations requests. In response, node x will receive N_a number of responses. So, the communication overhead will be $2(N_a)$. If node x wants to communicate with all other nodes (N) in the network, then the communication overhead is $2(N_a)(N-1)$. Further, if node x wants to communicate with BS in the h hop, then the number of acknowledgment generated is $2(h-1)$. When all N nodes want to communicate with BS, the total number of acknowledgment is $2N(h-1)$. Therefore, the total communication overhead for GATE is $2(N_a)(N-1) + 2N(h-1)$.

It is also worth mentioning that the number of neighbor $|N_a|$ is always less than the number of node in a cluster $|n|$.

7.3.2 Comparison

In this section, we have compared the communication overhead of our proposed scheme GATE with LDTS [11] and GTMS [12].

In LDTS, when node x from a cluster wants to communicate with BS through its cluster head (CH), the node x sends a maximum of one feedback request and also receives a maximum of one response. So, the communication overhead between a node and its CH is $(C_{\mathrm{LDTS(x-CH)}} = 2)$. When a CH of one cluster wants to communicate with another CH, it sends one feedback request and in turn receives one feedback response. Hence, communication overhead between any two CH is $(C_{\mathrm{LDTS(CH-CH)}} = 2)$. When CH of any cluster wants to collect feedback from its cluster members CM, it will send the maximum of the δ number of feedback request. In reply, it will receive a δ number of feedback responses. So, communication overhead between CH and its CM is $(C_{\mathrm{LDTS(CH-CM)}} = 2\delta)$. When all the CHs collect feedback from their CM, the communication overhead will be $2M\delta$. When BS wants to collect feedback from all the CHs, it sends the maximum of the M number of request and in turn will receive a M number of responses also. So, the communication overhead between BS and all CHs is $(C_{\mathrm{LDTS(BS-CH)}} =$

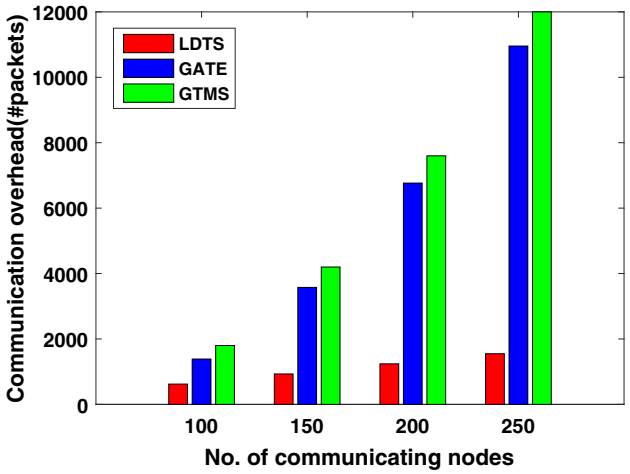

Fig. 9 Communication overhead

$2M$). If a node x that presents in a cluster wants to communicate with BS in h hops, then the communication overhead will be $(C_{\text{LDTS(x-CH)}} + C_{\text{LDTS(CH-CH)}}(h-2)) = 2+2(h-2) = 2+2h-4 = 2(h-1)$. If all N number of node want to communicate with BS, then the total communication overhead of LDTS is $2N(h-1) + 2M\delta + 2M$.

In GTMS, the total communication overhead is $2N(\delta + h-4)$ (for details about communication overhead of GTMS, please see [27]).

Figure 9 shows that the communication overhead increases with increase in the number of communicating node (N) in all three schemes. The communication overhead in GATE is less as compared to GTMS, but more as compared to LDTS. In GATE, when a node x wants to collect the feedback about another node y, the node x collects the feedback from all its trusted neighbors, but in LDTS node x collects feedback from its CH.

7.3.3 Memory Overhead Analysis

In GATE, each node maintains a transaction table to monitor and store the trust value of its neighbor nodes only. The different parameters on memory consumption and their corresponding memory requirements are shown in Table 2. The node id consumes 2 bytes of memory space. Each successful and unsuccessful interaction within a time unit(t_k) consumes 1 byte of memory space each, present in the sliding timing window (W). To store the direct trust, 0.5 bytes is required. Therefore, total memory space required to store a trust record in the transaction table is $2.5 + 2k$ bytes, where k is the number of time units in a sliding window (W).

Since in the worst case a node can have a maximum of the $N_a - 1$ number of neighbors, node can consume at most $(2.5 + 2k)(N_a - 1)$ bytes of memory space. Assuming that the minimum possible value of time unit k is one, the total memory overhead for GATE is $4.5(N_a - 1)$.

7.3.4 Comparison

This section presents the memory overhead comparison of GATE with LDTS and GTMS. In LDTS, the size of each trust record is 7 (refer [11]). As there are δ number of nodes presents in a cluster in average, the memory requirement at CM is $7(\delta - 1)$. Further, each CH in LDTS maintains two tables, one table(matrix) stores the feedback matrix, so the memory overhead is $0.5(\delta - 1)^2$. In another table, each CH maintains a trust record of 7 bytes. As there are M number of CHs in the network, therefore memory overhead for second table is $7(M - 1)$. In total memory overhead at CH level is $7(M - 1) + 0.5(\delta - 1)^2$. Hence, the maximum memory overhead for the entire network that consists of N number of nodes and M number of CH is $M_{\text{LDTS}} = 7N(\delta - 1) + M(7(M - 1) + 0.5(\delta - 1)^2)$.

In GTMS ([12]), the maximum memory overhead for the entire network that consists of N number of nodes and M number of CH is $M_{\text{GTMS}} = 8N(\delta - 1) + 8M(M + \delta - 2)$ (please see [27]).

Figure 10 shows the memory overhead of GATE, LDTS and GTMS with varying number of communicating node(N). It is evident from Fig. 10 that the memory overhead in GATE is comparatively very less than the both LDTS and GTMS.

8 Simulation-Based Analysis and Performance Evaluations of GATE

In this section, we evaluate the performance of GATE through simulation using MATLAB 2016b because our entire trust computation is started after gathering the data about number of data packet send, number of data packet received successfully, total number of communication happens between two nodes in a specified period, and number of control packet send successfully, etc. In Sect. 8.1, we have shown the effect of misbehavior on direct trust, and also we have compared our scheme with another related scheme like GTMS, LDTS and Ishmonav et al. [13] schemes. Further, in Sect. 8.5, we have shown the effect of dishonest recommendation on indirect trust computation and the comparison with other related schemes like GTMS and Ishmonav et al. [13]. In Sect 8.5, we have not compared our scheme with LDTS because in LDTS the indirect trust computation is based on the counting of some positive and negative feedback about the accessed node instead of actual feedback value by the honest recommender. The parameters used in the simulations are shown in Table 3.

Table 2 Memory overhead of each node

Memory consuming parameters		Memory size required
Node id		2 bytes
Number of successful transactions in each time unit (t_k) of sliding window (w)	t_1	1 byte
	.	.
	.	.
	t_k	1 byte
Number of unsuccessful transactions in each time unit (t_k) of sliding window (w)	t1	1 byte
	.	.
	.	.
	t_k	1 byte
Direct trust value		0.5 byte
Indirect trust value		Nil

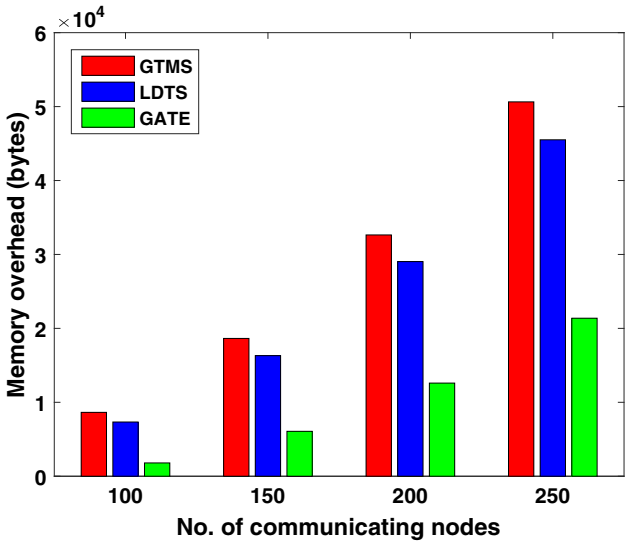

Fig. 10 Memory overhead

Table 3 Simulation parameters

Parameters	Value
Testbed dimension	500×500 m^2
Node deployment	Random
Number of node	600
Radio range	40 m
Trust value range	0–10
Initial trust value of node	5
Packet size	50 bytes
Packet interval	10 s
Initial node energy	30 J
Transmitter/receiver circuitry dissipation	0.5 nJ/bit
Data aggregation energy (EDA)	0.5 nJ/bit
TTL	4

Table 4 Simulation parameters to show the effect of misbehavior on direct trust

Parameter	Parameter value
Number of behavior in each time unit (t_k)	10
Sliding window length (W_L)	Minimum:3, maximum:7
Trust threshold	5
Trust aging factor (TF)	0.8 (for all types of trust scheme)

8.1 Effect of Misbehavior on Direct Trust in GATE and Other Related Schemes

To show the effect of misbehaviors (unsuccessful transactions) on direct trust, we generate 10 number of random behaviors in each time unit t_k. For each of the behavior, a random number is generated between 0 and 10 in each time unit t_k. If the generated random number is greater than or equal to 5, then the behavior is a good/ successful transaction. If the generated random number is less than or equal to 4, then the generated behavior is considered as a malicious/ unsuccessful transaction. By considering the following parameters shown in Table 4, we simulate the effect of misbehavior on direct trust in time window W. To compare with other schemes, we modified the trust domain 0 to 10 in GTMS and Ishmonav et al. [13]. In LDTS, the trust domain remains same.

Figure 11 shows the effect of malicious behavior on direct trust estimation. It is evident from the result that the direct trust value in our proposed direct trust computation scheme lies between GTMS and LDTS. Further, the direct trust value also lies below the Ishmonav et al. [13], i.e., in GATE, the resulted trust values are stricter because our scheme considers both the reward and penalty policies.

8.2 ON–OFF Attack Detection in GATE

As the ON–OFF attack manifested in data forwarding plane, in this section, we evaluate and compare our proposed direct

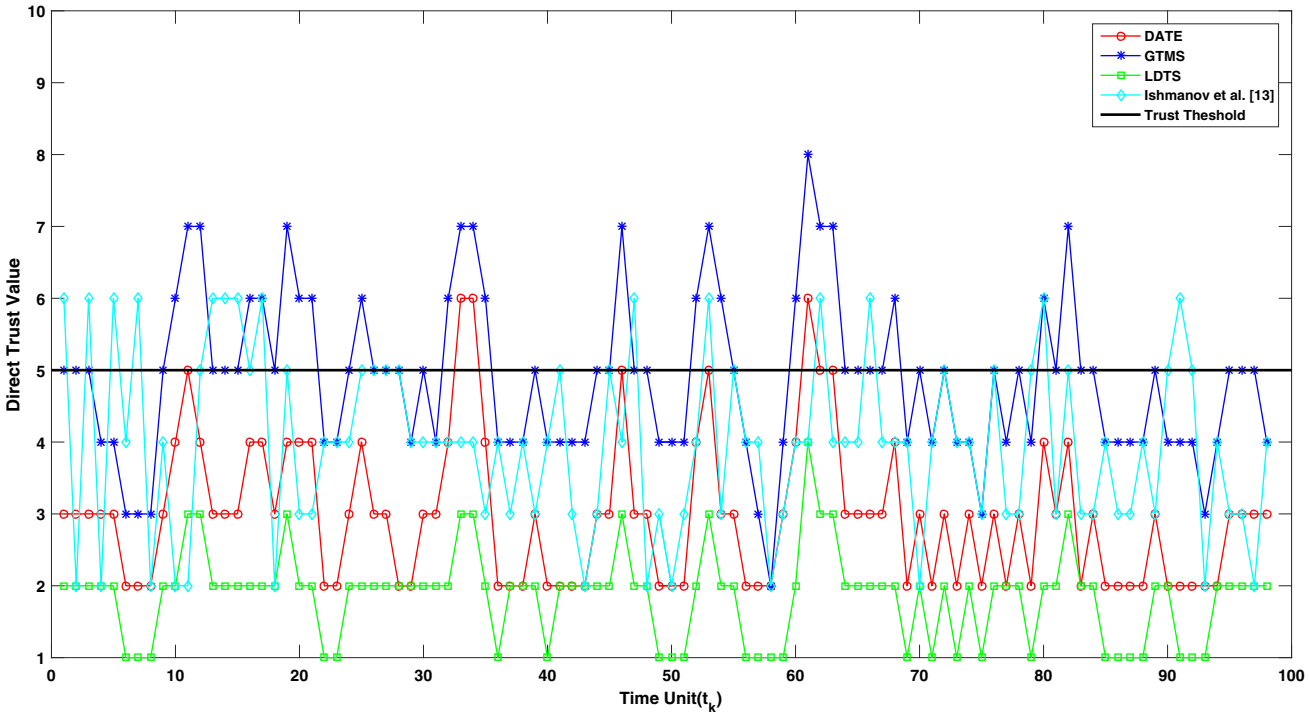

Fig. 11 Effect of malicious behavior on direct trust

trust scheme under the ON–OFF attacks behavior of malicious nodes. To make the simulation more reasonable, we use the following two different types of ON–OFF attacks:

- **Type 1 ON–OFF attack:** A malicious node exhibits more ON–OFF attack in a dynamic sliding window, i.e., the periodicity of attack (misbehavior) is more.
- **Type 2 ON–OFF attack:** A malicious node exhibits less ON–OFF attack in a dynamic sliding window as compared to type 1, i.e., the periodicity of attack (misbehavior) is less as compared to type 1.

Further, to simulate the ON–OFF attack detection under above discussed two scenarios, we use parameters in Table 5.

Figure 12 shows the ON–OFF attack detection rate in different trust scheme under two different types of ON–OFF attack. This simulation is evaluated under (90, 80, 70, 60%) and (40, 30, 20, 10%) periodicity of misbehavior (PM) when the sliding window length varies from 3 to 7 for both of type 1 and type 2 ON–OFF attack, respectively.

Subplot 1 of Fig. 12 shows that there is 100, 98, 95 and 95% detection rate whenever the periodicity of misbehavior is 90% (Type 1 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively. The detection rate is 80, 70, 67 and 68% whenever the PM is 40% (Type 2 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively.

Subplot 2 of Fig. 12 shows that there is 95, 87, 75 and 73% detection rate whenever the periodicity of misbehavior is 80% (Type 1 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively. The detection rate is 48, 30, 25 and 20% whenever the PM is 30% (Type 2 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively.

Similarly, subplot 4 of Fig. 12 shows that there is 89, 75, 73 and 71% detection rate whenever the periodicity of misbehavior is 60% (Type 1 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively. The detection rate is 10, 5, 3 and 3% whenever the PM is 10% (Type 2 ON–OFF attack) for GATE, Ishmonav et al. [13], LDTS and GTMS, respectively.

Further, it is evident from the simulation results that the detection rate is very high in all three schemes whenever the periodicity of misbehavior is more, and the detection rate is comparatively less when the periodicity of misbehavior is less. However, the detection rate under two different types of ON–OFF attack is more in GATE as compared to Ishmonav et al. [13], LDTS and GTMS.

8.3 Effect of Dishonest Recommendation on Indirect Trust Computation in GATE

To show the effect of dishonest trust recommendations, we conducted an experiment with and without using any filtering

Table 5 Simulation parameters for ON–OFF detection

Parameter used	Parameter value
Sliding Window Length (WL)	Minimum: 3 and maximum: 7. Changes dynamically for one more unit based on the periodicity of attack (i.e., number of good and number of misbehavior) in W
Trust threshold	5
Trust aging factor (TF)	0.8 (for all types of trust scheme)
Periodicity of off (i.e., number of off/good behavior) in window (W)	Randomly generated: 3–8, depending on W_L
Periodicity of on (i.e., number of on/bad behavior) in window (W)	Randomly generated: 3–8, depending on W_L
Threshold of periodicity of misbehavior (θ)	0.3

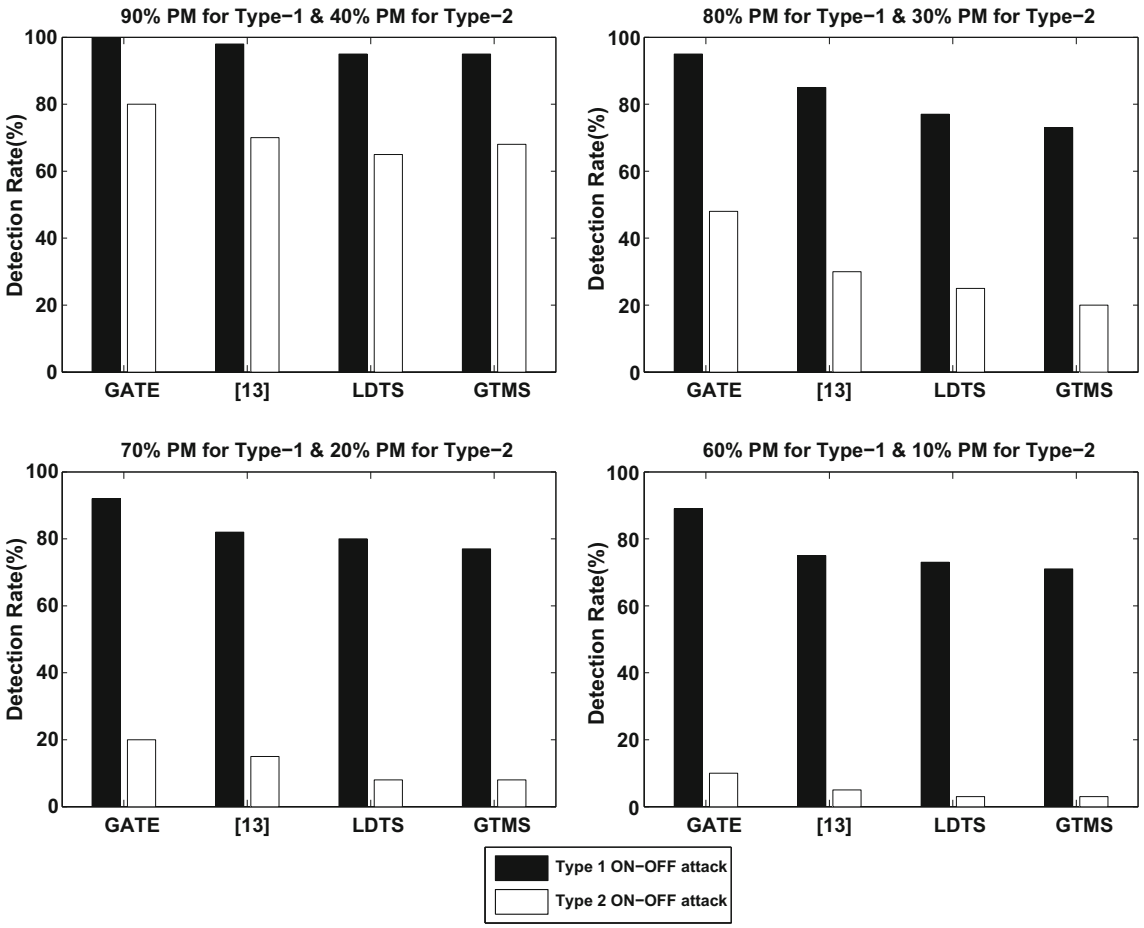

Fig. 12 ON–OFF detection rate

mechanism. In this experiment, we consider the following four scenarios as shown in Table 6.

Further, in this experiment whenever proposed filtering mechanism is used, Eq. (11) is considered to examine the indirect trust of the assessed node. On the other hand, the indirect trust of the assessed node is computed using Eq. (12) whenever no filtering mechanism is used, i.e., in the presence of dishonest recommender.

$$\mathrm{IT}_{x,y} = \frac{\sum_{i=1}^{m} \sqrt{\left(\mathrm{TE}_{x,\mathrm{RT}_i} \times \mathrm{RT}_{i,y}\right)}}{|m|} \tag{12}$$

where x is the accessing node; y is the accessed node. $\mathrm{TE}_{x,\mathrm{RT}_i}$ is the trust of assessing node x on each recommender RT; $|m|$ is the number of recommender.

Figure 13 shows the estimated indirect trust values over 100 simulation rounds under scenario 1 and scenario 2 with

Table 6 Evaluation of dishonest recommendation on indirect trust computation

Filter use	Scenario	Type of recommender	Type of accessed node
	Scenario 1	Honest	Benevolent
	Scenario 2	Honest	Malicious
	Scenario 3	Dishonest, without recommendation deviation	–
	Scenario 4 (i)	Dishonest, with recommendation deviation	Benevolent
	Scenario 4 (ii)	Dishonest, with recommendation deviation	Malicious

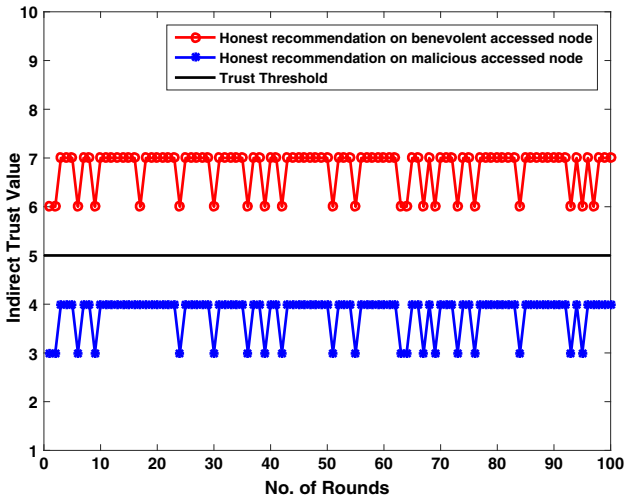

Fig. 13 Indirect trust value in the presence of honest recommender

filtering mechanism. For each indirect trust estimation, the direct trust value of the honest neighbor recommender on benevolent assessed node randomly generated between 5 and 10, whereas the direct trust value of the honest neighbor recommender on malicious assessed node randomly generated between 0 and 4. As Fig. 13 depicts, in our indirect trust computation model the estimated value of indirect trust of the benevolent assessed node is always above the trust threshold, whereas the indirect trust value of the malicious assessed node is always below the trust threshold.

Figure 14 illustrates the estimated indirect trust values under scenario 3 without employing any filtering mechanism and recommendation deviation. As Fig. 14 shows, with 10, 20 and 30% dishonest recommender (DR) the indirect trust of the node is not distorted. However, the resilience of our proposed model degraded when the dishonest recommender percentage increases to 40–50%.

8.4 Detection Ratio of GATE and Related Scheme

To determine the percentage of the dishonest recommender, we evaluate our proposed GATE scheme with different percentage of dishonest recommendations and the different threshold value of variation factor C (as per Eq. (10)). Figure 15 depicts that, with the threshold value of variation factor $C = 1$, the dishonest recommender detection rate is 100 and 9% in the best case and worst case, respectively. Similarly, when the threshold $C = 2$, the detection rate is 82% in the best case and 2% in worst case. Moreover, the results that are shown in Fig. 15 are correlated with Fig. 14, i.e., when the percentage of dishonest recommender increases to 40 and 50%, the rate detection is very poor.

8.5 Comparative Analysis of GATE with Related Scheme

Figures 16, 17, 18 and 19 demonstrate the estimated indirect trust (IT) value in the presence of 10, 20, 30 and 40% dishonest recommender, respectively. As shown in Figs. 16 and 17, with 10 and 20% dishonest recommender, the estimated value of the indirect trust is remained undistorted in our proposed GATE model, in GTMS and Ishmonav et al. [13], i.e., in all three models the computed indirect trust value in each round is on above trust threshold. However, Fig. 18 depicts that the value of computed indirect trust in each round is distorted less with 30% dishonest recommender in GATE as compared to GTMS and Ishmonav et al. [13], i.e., the computed indirect trust value in GATE lays less below the trust threshold as compared to GTMS and Ishmonav et al., and Fig. 19 also gives the same conclusion as that of Fig. 18 with 40% dishonest recommender.

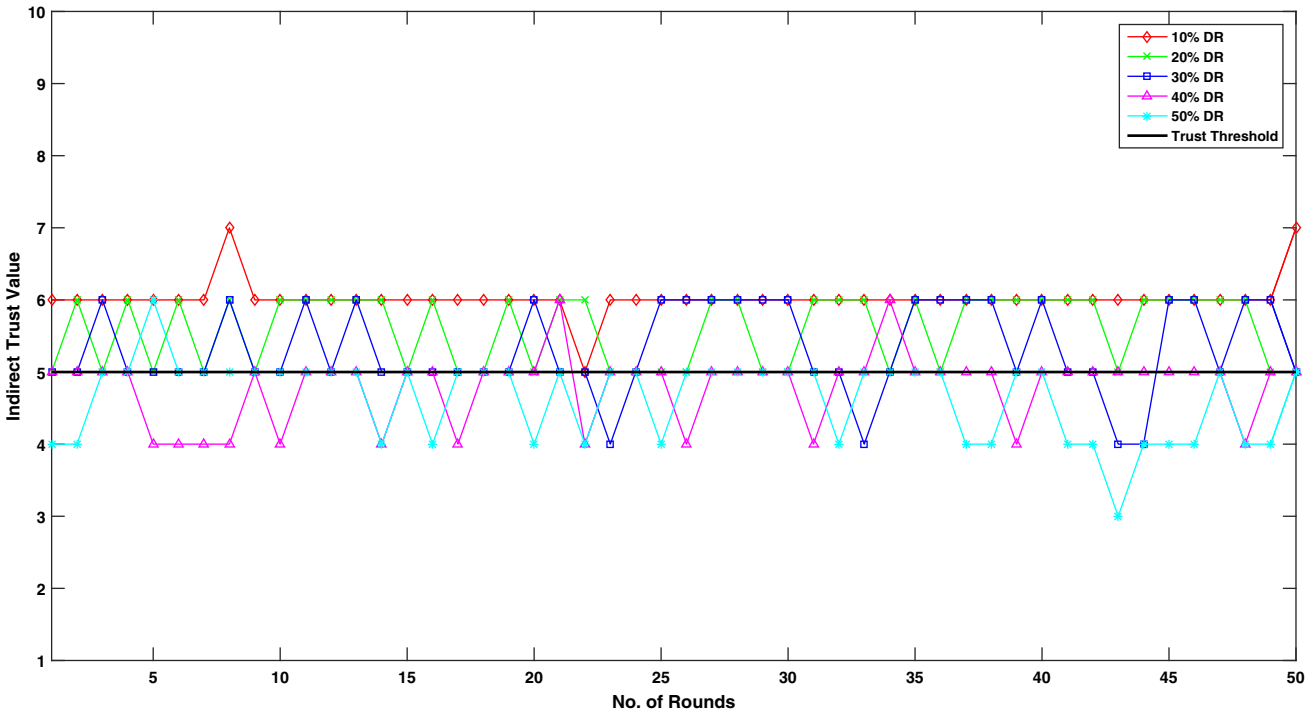

Fig. 14 Indirect trust values in the presence of dishonest recommender

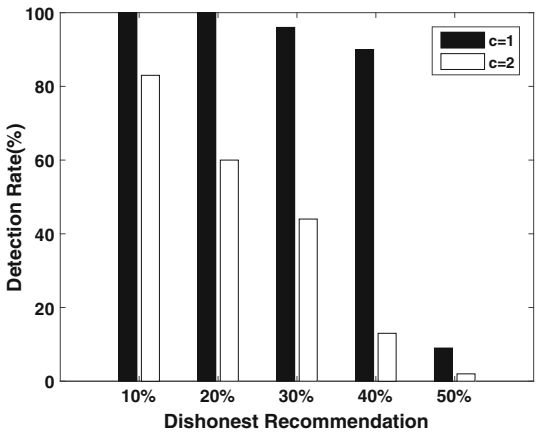

Fig. 15 Detection rate of dishonest recommender

8.6 Impact of Dishonest Recommender Launches Bad Mouthing and Ballot Stuffing Attack

In order to evaluate the performance of GATE under scenario 4(i) (i.e., badmouthing attack) and scenario 4(ii) (i.e., ballot stuffing attack), we use the following two factors:

1. Percentage of dishonest recommender
2. Recommendation deviation ($\alpha\%$): defined as the percentage of deviation in recommendation value given by recommender from actual indirect trust value of the assessed node.

In the case of bad mouthing attack, the recommender always gives less trust value than the actual trust value of the accessed node. Therefore, if T is the actual trust value of the assessed node and $\alpha\%$ is the recommendation deviation, then the recommendation given by the bad mouthing attacker and ballot stuffing attacker is given by Eqs. (13) and (14), respectively.

$$\mathrm{RT} = T - (T * \alpha\%) \tag{13}$$

$$\mathrm{RT} = T + (T * \alpha\%) \tag{14}$$

Let us assume that an accessed node x is benevolent with $T = 7$ and accessed node y is malicious with $T = 4$. Node x suffers in bad mouthing attack and node y suffers in ballot stuffing attack. Figures. 20 and 21 illustrate the effect of bad mouthing and ballot stuffing attacking nodes on the accessed node x and node y, respectively. The indirect trust of node x and node y is computed by using Eq. (12) with varying percentage of dishonest recommendations (DR) 10 and 40% and recommendation deviations varying between 20 and 80%. As Fig. 20 shows, with a low percentage (i.e., 10%) of DR and recommendation deviation (i.e., 20%) the estimated indirect trust value of accessed node x is not distorted much (i.e., the estimated indirect trust value conferred the accessed node x as benevolent). However, with a high percentage (i.e., 40%) of DR and recommendation deviation (i.e., 80%) the estimated indirect trust value of assessed node x conferred the node x as malicious, which is grimly disagree with the actual trust value of node x.

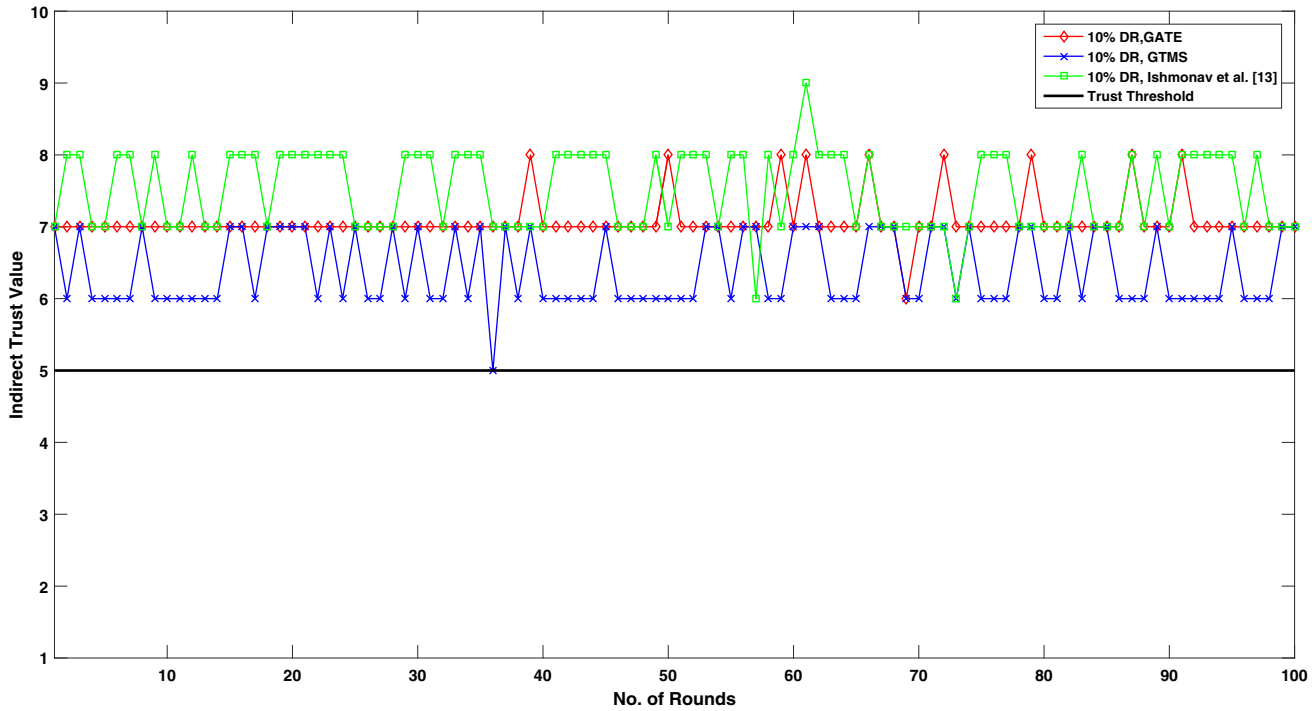

Fig. 16 IT with 10% DR in GATE, GTMS and Ishmonav et al. [13]

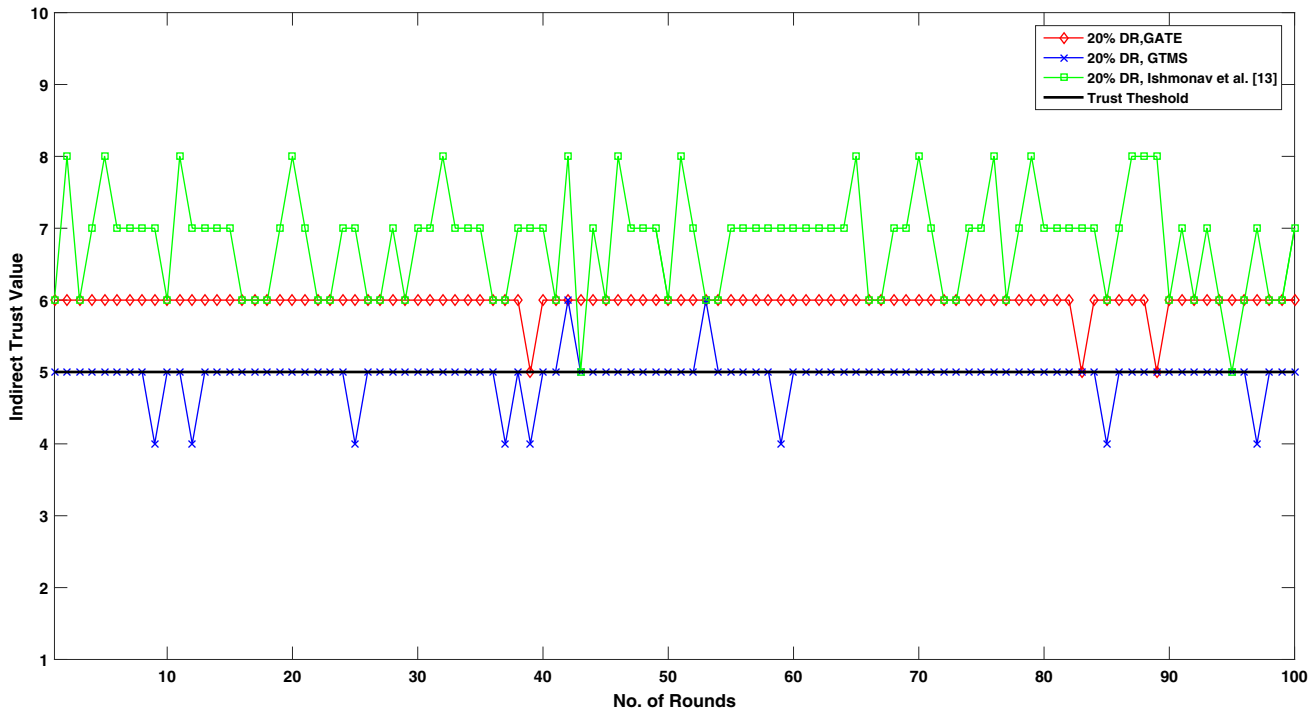

Fig. 17 IT with 20% DR in GATE, GTMS and Ishmonav et al. [13]

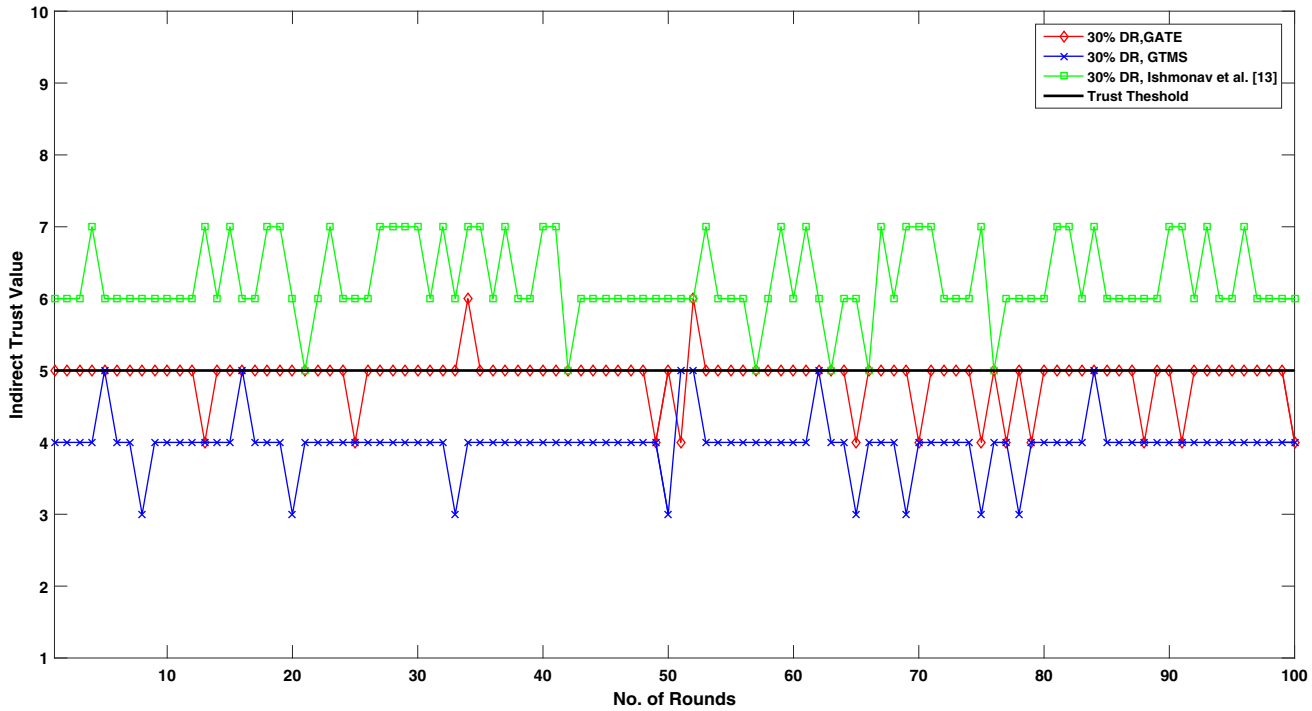

Fig. 18 IT with 30% DR in GATE, GTMS and Ishmonav et al. [13]

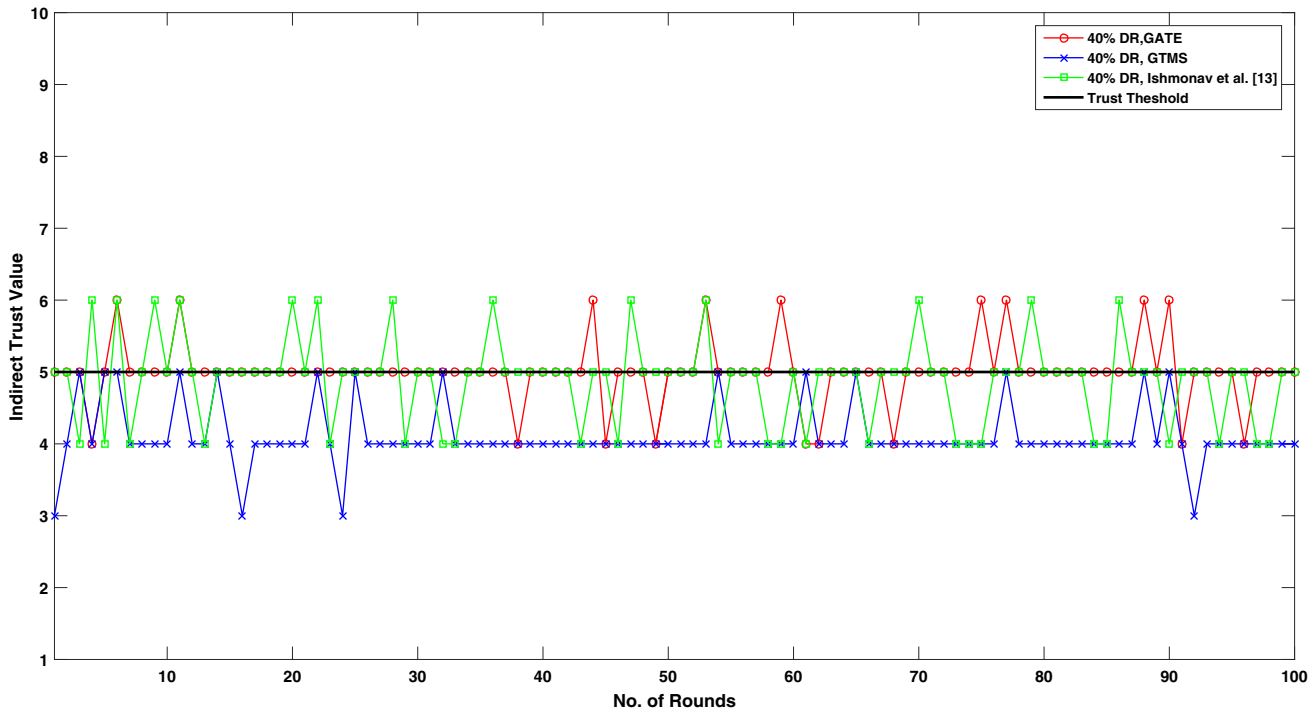

Fig. 19 IT with 40% DR in GATE, GTMS and Ishmonav et al. [13]

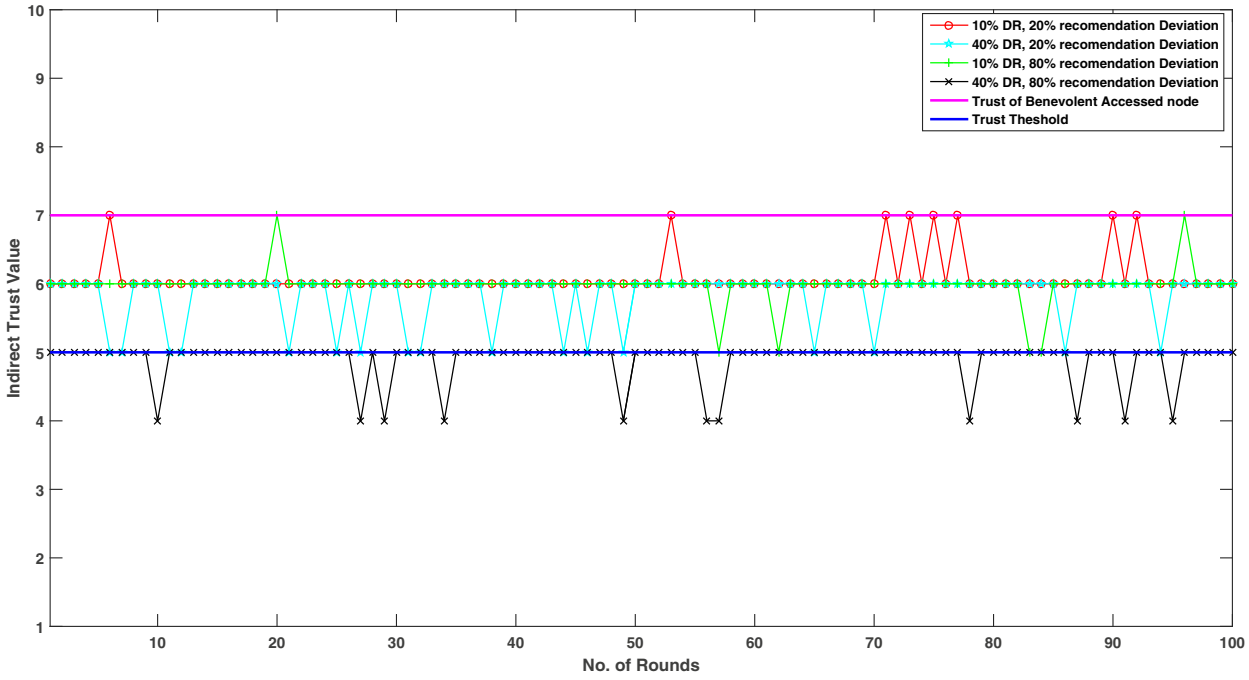

Fig. 20 Effect of bad mouthing attack

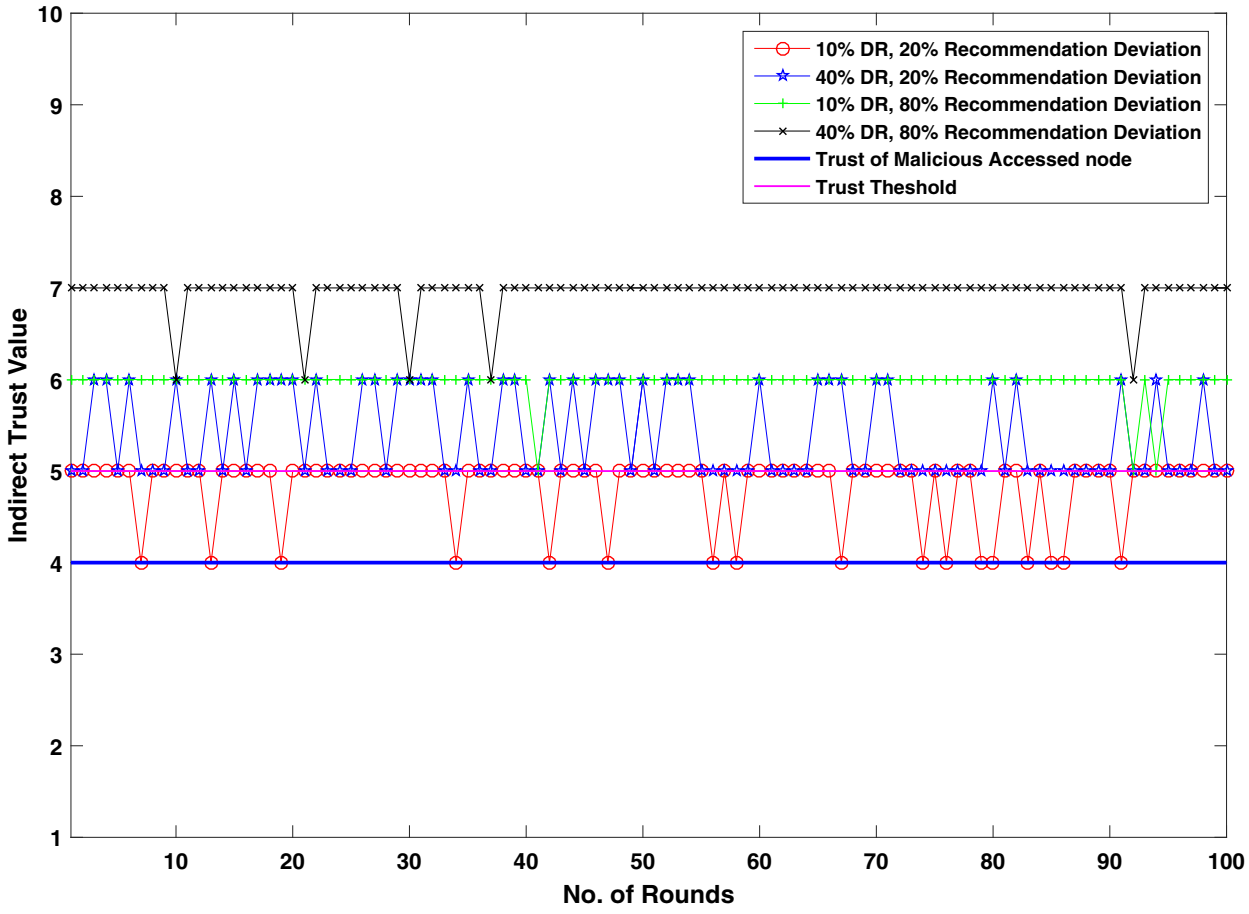

Fig. 21 Effect of ballot stuffing attack

Figure 21 shows with a low percentage (i.e., 10%) of DR and recommendation deviation (i.e., 20%) the assessed node y is conferred as same with its actual behavior. However, with 40% DR and 80% recommendation deviation the estimated indirect trust value is inferior, i.e., the estimated indirect trust of assessed node y, has completely digressed from its actual trust value.

An imperative observation from Figs. 20 and 21 is that with a high percentage of dishonest recommender (i.e., bad mouthing and ballot stuffing attackers) and high recommendation deviations in the absence of a filtering mechanism the trust management system is more vulnerable. Another important affirmation is that low percentage of DR with high recommendation deviation has almost same effect as that of a high percentage of DR with low recommendation deviation, which is evident from 10% DR with 80% recommendation deviation and 40% DR with 20% recommendation deviation.

9 Conclusion and Future Work

In this paper, the proposed a trust-based defense mechanism GATE provides a guard to the trust management system from various kinds of internal attacks. The proposed lightweight direct trust estimation method includes both the weight to past behavior and periodicity of misbehavior to improve the defense of trust management system itself. Furthermore, direct trust establishment scheme uses both reward and penalty policies for each good and malicious behavior of a node, respectively, to make the trust computation more realistic. The consideration of positive integer based trust domain and dynamic timing window demands fewer system resources. It also helps the system to segregate between ON and OFF attack nodes and transient malicious nodes by using trust regain scheme. However, finding the optimal value of sliding window length is one of the challenges in GATE, which we shall consider in our future work.

In indirect trust computation, the selection of appropriate recommender is based on variation factor to guard the trust management system against bad mouthing and ballot stuffing attackers. Furthermore, to filter out the dishonest recommendation, the proposed scheme use both own experiences of accessing node and majority view of the recommender, based on MADAM. Variation factor is used to filter out honest recommendations from dishonest one. It is also evident from the theoretical and simulation-based analysis that the effectiveness of the proposed scheme is resilient against bad mouthing and ballot stuffing attackers. Further, it is ascertained that our proposed scheme has better performance to detect dishonest recommendation than GTMS, and resilience of our scheme degrades when the dishonest recommendation increases to 50%.

References

1. Wang, Y.; Attebury, G.; Ramamurthy, B.: A survey of security issues in wireless sensor networks. IEEE Commun. Surv. Tutor. **8**(2), 2–23 (2006)
2. Osman, K.; Samee, U.K.; Sajjad, A.M.; et al.: Comparative study of trust and reputation systems for wireless sensor networks. Secur. Commun. Netw. **6**(6), 669–688 (2013)
3. Lopez, J.; Roman, R.; Agudo, I.; Fernandez-Gago, C.: Trust management systems for wireless sensor networks: best practices. Comput. Commun. **33**(9), 1086–1093 (2010)
4. Lu, Y.; Lin, K.; Li, K.: Trust evaluation model against insider attack in wireless sensor networks. In: IEEE Second International Conference on Cloud and Green Computing (CGC), pp. 319–326 (2012)
5. Yan, S.; Zhu, H.; Ray, L.K.J.: Defense of trust management vulnerabilities in distributed networks. IEEE Commun. Mag. **46**(4), 112–119 (2008)
6. Momani, M.; Challa, S.; Aboura, K.: Modeling trust in wireless sensor networks from the sensor reliability perspective. In: Innovative Algorithms and Techniques in Automation, Industrial Electronics and Telecommunications, pp. 317–321. Springer, Netherlands (2007)
7. Anita, X.; Bhagyaveni, M.A.; Martin Leo Manickam, J.: Collaborative lightweight trust management scheme for wireless sensor networks. Wirel. Pers. Commun. **80**(1), 117–140 (2015)
8. Chae, Y.; DiPippo, L.C.; Sun, Y.L.: Trust management for defending on-off attacks. IEEE Trans. Parallel Distrib. Syst. **26**(4), 1178–1191 (2015)
9. Chen, X.; Makki, K.; Yen, K.; Pissinou, N.: Sensor network security: a survey. IEEE Commun. Surv. Tutor. **11**(2), 52–73 (2009)
10. Ishmanov, F.; Kim, S.W.; Nam, S.Y.: A robust trust establishment scheme for wireless sensor networks. Sensors **15**, 7040–7061 (2015)
11. Li, X.; Zhou, F.; Du, J.: LDTS: a lightweight and dependable trust system for clustered wireless sensor networks. IEEE Trans. Inf. Forensic Secur. **8**, 924–935 (2013)
12. Shaikh, R.A.; Jameel, H.; dAuriol, B.J.; Lee, H.; Lee, S.Y.; Song, Y.J.: Group-based trust management scheme for clustered wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. **20**(11), 1698–1712 (2009)
13. Ishmanov, F.; Kim, S.W.; Nam, S.Y.: A secure trust establishment scheme for wireless sensor networks. Sensors **14**, 1877–1897 (2014)
14. Chen, S.; Zhang, Y.; Liu, Q.; Feng, J.: Dealing with dishonest recommendation: the trail in reputation management court. Ad Hoc Netw. **10**, 1603–1618 (2012)
15. Qu, C.; Ju, L.; Jia, Z.; Xu, H.; Zheng, L.: Light-weight trust based on-demand multi path routing protocol for mobile ad hoc networks. In: 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-13), pp. 42–49 (2013)
16. Sahoo, R.R.; Singh, M.; Sardar, A.R.; Mohapatra, S.; Sarkar, S.K.: TREE-CR: Trust based secure and energy efficient clustering in WSN. In: Emerging Trends in Computing, Communication and Nano Technology (ICE-CCN), IEEE International Conference, Tirunelveli, pp. 532–538 (2013)
17. Momani, M.; Challa, S.: Survey of trust models in different network domains. Int. J. Ad Hoc Sensor Ubiquitous Comput. **1**(3), 1–19 (2010)
18. Momani, M.; Agbinya, J.; Navarrete, G.P.; Akache, M.: A new algorithm of trust formation in wireless sensor networks. In: First IEEE International Conference on Wireless Broadband and Ultra Wide-band Communications; Sydney, Australia (2006)
19. Bao, F.; Chen, I.; Chang, M.; Cho, J.: Hierarchical trust management for wireless sensor networks and its applications to

trust-based routing and intrusion detection. IEEE Trans. Netw. Serv. Manag. **9**(2), 169–183 (2012)
20. Zahariadis, T.; Leligou, H.C.; Trakadas, P.; Voliotis, S.: Trust management in wireless sensor networks. Eur. Trans. Telecommun. **21**, 386–395 (2010)
21. Sahoo, R.R.; Sardar, A.R.; Singh, M.; Ray, S.; Sarkar, S.K.: A bio inspired and trust based approach for clustering in WSN. Nat. Comput. **15**(3), 423–434 (2016)
22. Zhan, G.X.; Shi, W.S.; Deng, J.L.: Sensor Trust: a resilient trust model for wireless sensing systems. Pervasive Mob. Comput. **7**(4), 509–522 (2011)
23. Luo, W.; Ma, W.; Gao, Q.: A dynamic trust management system for wireless sensor networks. Secur. Commun. Netw. **9**(7), 613–621 (2015)
24. Crosby, G.V.; Pissinou, N.; Gadze, J.: A framework for trust-based cluster head election in wireless sensor networks. In: Proceedings of Second IEEE Workshop on Dependability and Security in Sensor Networks and Systems (2006)
25. Guowei, W.; Zhuang, D.; Yibo, H.; Jung, T.; Fiore, U.; Yim, K.: A dynamic trust model exploiting the time slice in WSNs. Soft. Comput. **18**, 1829–1840 (2014)
26. Huber, P.J.: Robust Statistics. Wiley, New York (1981)
27. Anita, X.; Martin, L.M.J.; Bhagyaveni, M.A.: Two-Way acknowledgment-based trust framework for wireless sensor networks. Int. J. Distrib. Sens. Netw. **9**(5), 1–14 (2013)