Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 796 – 815

doi:10.3906/elk-2003-5

Dynamic distributed trust management scheme for the Internet of Things

Syed Wasif Abbas HAMDANI[1], Abdul Waheed KHAN[2,*], Naima ILTAF[1], Javed Iqbal BANGASH[3], Yawar Abbas BANGASH[1], Asfandyar KHAN[3]

[1]Department of Computer Software Engineering, National University of Sciences and Technology, Islamabad, Pakistan

[2]FAST School of Computing, National University of Computer and Emerging Sciences, Islamabad, Pakistan

[3]Institute of Computer Sciences and Information Technology, The University of Agriculture, Peshawar, Pakistan

Received: 01.03.2020 • **Accepted/Published Online:** 25.08.2020 • **Final Version:** 30.03.2021

Abstract: The Internet of Things (IoT) comprises of a diverse network of homogeneous and heterogeneous nodes that can be accessed through network ubiquitously. In unattended environments, the IoT devices are prone to various attacks including ballot-stuffing, bad-mouthing, self-promotion, on-off, opportunistic behavior attacks, etc. The on-off attack is difficult to detect as nodes switch their behavior from normal to malicious alternatively. A trust management model is a tool to defend the IoT system against malicious activities and provide reliable data exchange. The majority of existing IoT trust management techniques are based on static reward and punishment values in pursuit of trust computation thereby allowing the misbehaving nodes to deliberately perform on-off attacks. Due to the static nature of awarding scores, these schemes fail to identify malicious nodes in certain cases. In this paper, a dynamic and distributed trust management scheme (DDTMS) is proposed where nodes can autonomously evaluate peer nodes' behavior and dynamically grant reward and penalty score. The proposed scheme successfully detects the on-off attack and isolates the misbehaving nodes thereby classifying them into three distinct categories based on their severity levels i.e. low, mild, and severe. Simulation-based performance evaluation shows improved performance of the proposed DDTMS against other state-of-the-art schemes thereby requiring less time and fewer interactions for successfully identifying malevolent behavior of compromised nodes.

Key words: Internet of Things, trust management, distributed trust management, on-off attacks, DDTMS

1. Introduction

The Internet of Things (IoT) [1] is the interconnection of homogeneous and heterogeneous physical objects which are attached to the internet and can exchange data [2, 3]. With the advancements being made in information and sensors technology, IoT is widely used in numerous applications, such as health care, defense systems, home and industrial automation, supply chain, environmental system monitoring etc. [4, 5]. In IoT, "things", refer to any physical device, equipped with processing and communication modules thereby qualifying to be considered as smart devices [6, 7]. These things referred as nodes from here onwards, are autonomous in nature thereby collecting data from their surroundings through sensors and exchanging data with their peers whenever required. The IoT nodes are comprised of limited resources i.e. energy, processing, communication and computation capabilities. Therefore, such limitations create challenging issues in providing security and are vulnerable to various types of attacks and risks. Security gets further crucial when the systems have to operate in

*Correspondence: awk_career@yahoo.co.uk

unattended and hostile environments. The sensed data must be protected to ensure integrity and confidentiality. The traditional internet-based security protocols can not be directly adapted in IoT domain due to heterogeneity of devices as well as their stringent resources. In some cases, constrained nodes within the network usually need the support of other nodes to complete a specific task. Although there are numerous issues that are associated with the IoT implementation, however, Security is of prime importance due to the autonomous nature of this technology. Trust plays a pivot role to secure a system which is comprised of multiple cooperating entities. Trust computation is not a one-time job and a device may start misbehaving even after certain security and privacy measures are in place. Therefore, the behavior of devices in a distributed environment need to be monitored and analyzed on a continuous basis. Trust management offers such behavioral analysis of devices, based on their past and current behavior, and/or recommendation [8]. It permits the coordination among devices subject to qualifying the trust value. Therefore, an object must choose trustworthy partners to avoid malevolent behavior [9, 10]. Trust management offer solutions to issues like identity management, enhanced user privacy and information security. In a thing-based service-oriented architecture, three main actors interact with each other directly namely service requester, service registry and a service provider [11]. As an example of services; in an air pollution controller system, the services are hosted on various sensor nodes (devices): a nitrous oxide (N_2O) sensor (sensor node 1), a carbon monoxide (CO) sensor (sensor node 2), sulphur dioxide (SO_2) sensor (sensor node 3) and a particle chemical composition (PM_{10}) sensor (sensor node 4). The overall purpose of such a multiservice IoT system would be sabotaged if some sensors start misbehaving in the form of ballot-stuffing, bad-mouthing, self-promotion, on-off, opportunistic behavior attacks [12] etc.

In IoT, on-off attack is comprised of two states i.e. on-state and off-state. The attack state is considered as on-state where the attacker stops providing requested service(s). In off-state, the node behaves like a normal node offering all services requested. The attacker node switches between these two states in on-off attack. In this regard, trust management plays key role of granting and revoking the trust score. Existing trust management schemes can be categorized into two groups namely centralized and distributed. In the centralized approach, trust information about nodes is stored on a central trust management server. Trust information is computed on demand when a node requests, service is delivered to the node at that moment. Trust management server is responsible to manage service request, evaluate service, and compute trust. Decision making is done on the server side [13]. In a distributed approach, nodes autonomously evaluate the trust score of peers and maintains trust tables comprising of node id, service list, and node trust of their neighbor nodes [14]. Trust information is computed after each transaction, updated in the trust table and propagated to the neighbor nodes.

Several distributed trust management models have been proposed in the literature; however, most of these contributions suffer from static reward and punishment strategy, thus allowing the malicious node(s) to deliberately execute on-off attacks in certain scenarios without affecting node(s) trust value. As a result, the on-off attacks carried out by malicious nodes remain undetected thereby badly affecting the operation of IoT network. In this paper, a dynamic distributed trust management scheme (DDTMS) is proposed where nodes can autonomously evaluate peer nodes' behavior and dynamically grant reward and penalty scores. The proposed scheme successfully detects the on-off attack and isolates the misbehaving nodes thereby classifying them into three distinct categories based on their severity levels i.e. low, mild, and severe. Once the trust score of a node reaches to severe category, the requesting node not only suspends its interaction with that malicious node but also shares this information with its peers in order to help them in their trust adjustment. This strategy helps in successfully identifying the on-off attack behavior of a malicious node in a variety of scenarios. Moreover, the proposed DDTMS require fewer interactions in detecting the malevolent behavior of nodes.

The rest of this paper is structured as follow: Section 2 provides an overview of existing trust management schemes. Section 3 provides the explicit details of the proposed dynamic distributed trust scheme including service request, service evaluation, trust computation and update. Section 4 demonstrates the simulation and validation of proposed model. Performance of the proposed scheme is analyzed in Section 5. At the end, Section 6 reveals the conclusion and identification of future research directions of the proposed scheme.

2. Literature review

At the core of IoT is wireless sensor networks (WSNs) where tiny sensor devices equipped with computing and communication modules sense their surrounding environments and cooperatively disseminate data for further analysis. We provide a critical analysis of existing trust management work being proposed for WSN and IoT in this section. Many of the existing contributions are based on centralized trust management whereas few schemes are of distributed nature. In [13], the authors presented a multiservice and context-aware approach based on a centralized approach that highlights the ability to discourage common attacks intended to aim the IoT environment. A centralized trust management server is responsible to manage service requests, decision making, and store nodes information. Authors identified nodes' dynamic status and recommended various metrics for the computation of nodes' trust level including social cooperativeness of node, its community interest and the difference in the indirect-interactions (recommendations). However, its weighting factor to compute trust using various capabilities of the node and social cooperation is practically difficult and thus identifying the malicious node. In [14], a distributed static IoT trust management scheme is proposed to detect and mitigate on-off attacks. The authors proposed a framework to discover the spiteful behavior of sensor nodes. It prevents possible on-off attacks in a multiservice environment. Although it successfully detects on-off attack in certain scenarios but due to its static reward and punishment strategy fails to identify malicious behavior in several cases. CORE [15] is a general trust-based technique that aims to identify the selfish behavior of nodes. This allows positive witnesses only to be broadcasted in the network. Therefore, it disregards the scenario where nodes join together by broadcasting false-positive facts to improve reputation scores (ballot-stuffing attack) of peer nodes. The reputation-based framework for sensor networks (RFSN) [16] is the earliest proposed trust model for WSNs. It monitors communications between sensor nodes. It relies on the trustworthiness score to defeat the ballot-stuffing attack of the observer node in order to weight in reports. However, this model is unable to detect the on-off attack in addition to the malicious node. In [17], an agent-based trust model is presented where the trustworthiness of every node in the network is evaluated by the mobile agents. It recommends shifting of storage and heavy computational tasks to dedicated agents instead of constrained nodes. Nonetheless, the computation of trustworthiness based on such a reputation model, it is problematic to differentiate a malevolent node from an assisting node. In [18], the authors purposed a model of repeated games to identify the malevolent nodes in the WSN. Trusted node predictability is done by game model and collaborative filtering. Every node preserves a particular rating of its neighbor node and trust is judged based on the rating. However, models based on game theory may not appropriate to determine entirely trust problems in the WSNs. Authors in [19], proposed a fuzzy logic-based trust model for the wireless sensor network. This model requires a well-defined and analytical scheme to define the context, trustors' goal and trust aggregation technique which takes the trustors' attitude into account towards received opinions and an explicit scheme for efficient trust decision making. In [20], the authors proposed a redemption and trust management scheme which can differentiate between malicious behaviors and temporary errors using the predictability trust to counter on-off attacks. However, if the overall behavior of the node is good then this model failed to identify the malicious

node. Authors in [21] proposed an extended version of the distributed static trust management model in IoT. Node's trust value is calculated based on the direct observations and recommendations. In the direct interactions scheme, a fixed score is granted for successful and failed transactions. This results in enabling the malicious node to deliberately misbehave in certain cases and thus such behavior cannot be detected as malicious. In [22], authors presented a distinctive trust management model namely HEXAGON which is based on the human concept of trust within the computational algorithm. Trust is computed considering reputation, privacy, peer recommendations, operational risk, operational cost, identity management, based on inference engine using fuzzy logic. This trust management model is very application specific and cannot be directly used in a broad range of applications. Moreover, this model does not consider any potential trust-related attack in such a critical and sensitive health care IoT environment. In [23], the authors presented a generic IoT trust management model considering certain quantitative and qualitative parameters. In this framework, trust computation is based on statistical, probabilistic, machine learning and fuzzy logic models. The limitation of this model is that its context is never taken into account and trust decision is done on single feedback. In [24], a multidimensional fuzzy logic-based IoT trust-aware access control (TACIoT) model has been proposed. Authorization decisions are based on nodes' trust values considering the reputation, social relationships, quality of service and security considerations. Nevertheless, decisions are limited to particular assumptions such as the identified ownership and feedback accessibility. Also, the correctness of the trust quantification model is another issue. Authors proposed a distributed trust scheme in [25], that selects feedback from a collaborating filtering approach based on social contact and friendship using a Bayesian probability method. It focuses on the application of service composition, scalable and adaptive trust management to assist integrated applications in service-oriented IoT systems. Nevertheless, the major issue with this model is the irrelevance of trust score to the context as it focused on user contentment practices. In [26], the authors proposed a trust model based on social similarity. Trust is computed based on social similarity for indirect trust aggregation. Nonetheless, one of the major concerns of a node's trust with its peer node is based on their association, instead of a common reputation that is derived from the node's public belief. In [27], another trust management model has been proposed based on predictability trust and a dynamic sliding window for the WSNs. The dynamic sliding window comprises good behavior and bad behavior window to track node's good and bad behavior respectively, which results in the prediction of the node's trust. Nevertheless, in the presence of more good past behaviors the overall trust computation is definitely a challenge and thus detection of on-off attacks. In [28], a Dirichlet-based scheme of trust management is proposed which focuses on the weighted sum of direct computed, past computed and indirectly computed trust values. The calculation of direct trust is based on the ratio of successful interactions to uncertain interactions while malevolent actions are considered double. In the case of more malicious transactions, the trust value of the node is degraded by defining a precaution factor. Nevertheless, if such a strategy is used carelessly, this can lead to the indictment of cooperating nodes causing random failures and also making the trust system more vulnerable to bad-mouthing attacks. Authors proposed robust trust management distributed model for IoT in [29]. Trust computation is based on direct observations and past recommendations. Devices can distribute the evaluated trust to the peer devices. Moreover, trust evaluation is based on various trust attributes including knowledge (integrity, feedback), experience (recommendation, competence) and reputation (reliability, honesty, cooperation) evaluations. Nevertheless, trust in the accuracy and consideration of specific contexts are great challenges. Also, certain trust-related attacks are even overlooked. In [30], a trust establishment scheme has been proposed to defend against on-off attacks for WSNs based on the five components i.e. reward, lenient penalty, severe penalty, misbehavior frequency, and sliding window. The authors considered trust values as

the positive integer with a range from 0 to 10 and defined 5 as an initial trust value of each node. However, with these positive values it is difficult to distinguish reward and penalty scores. The estimation of trust value comprises many parameters that are not defined including the threshold level. Also, misbehavior frequency and sliding window components not even considered while computing reward and penalty. In [31], the authors proposed a trust management scheme for the social internet of things using a clustering architecture that is based on the similarity of interest. Each cluster is led by a trusted admin and comprises of numerous objects having the similar interests with the assumption that every node within the cluster must transmit the trust value of the other community members to the admin to update the previous trust values. Nevertheless, gathering a lot of information can drain resources of a node since it needs higher computation capability, high communication costs to transfer recommendations, and larger space to store the trust information.

In the existing IoT trust management schemes and frameworks, there is a limited opportunity for forgiveness and recovery to the erroneous node(s). One major limitation is the fixed score awarded to compute reward and punishment, which allows the misbehaving node to deliberately perform on-off attack(s) in certain cases without affecting its trust value. The proposed model particularly addresses this issue, by awarding a dynamic score for reward and punishment after every transaction according to the behavior of each node. Time and number of interactions to detect malicious node is another limitation in the static models which took a long time and plenty of interactions to identify malevolent behavior of a compromised node whereas the proposed dynamic model efficiently detects such behavior in significantly less time and fewer interactions.

3. Proposed dynamic distributed trust management scheme

Comprehensive detail of the proposed scheme is provided which makes use of a dynamic distributed trust management scheme (DDTMS) to calculate the trust value of a node by computing reward and penalty scores dynamically. Whenever a node provides requested services, its trust score increases whereas trust score decreases, and the node is punished in case the node misbehaves and rejects the service(s) requested. The dynamic distributed trust management system is divided into five phases: initial communication, service request, service evaluation, trust computation, and finally trust update including trust dissemination. An overview of all these five phases is given as follows: Initially, in the first phase network initialization is performed where nodes after deployment exchange information with each other. This exchange of information is performed in the form of single-hop broadcasts with peer nodes which not only helps in neighbor discovery but also populates entries in their trust tables. The second phase describes the service request model. In the third phase, service response is evaluated and assigned reward OR punishment values. In the computation phase, the node's trust value is computed based on reward and punishment score(s). Finally, the newest computed node's trust value is updated in the requester node's trust table. Based on the node interaction with a malicious node, its behavior may fall in either low, mild, or severe category. The process flow diagram of the proposed DDTMS is shown in Figure 1.

Phase 1: initial communication

In the first phase, each node announces its presence by sending packets to the peer node(s) belonging to the same transmission range. With the help of these announcements, nodes discover their neighbors and get to know about the services they provide. At this stage, default trust values of service provider nodes are also populated in nodes trust table. Every node in the network offers multiple services. Since the nodes are dispersed randomly across the network, therefore, a node may or may not have node(s) within its radio range. Every node creates and maintains its own trust table which contains information about the neighbor node(s) i.e. node id, service

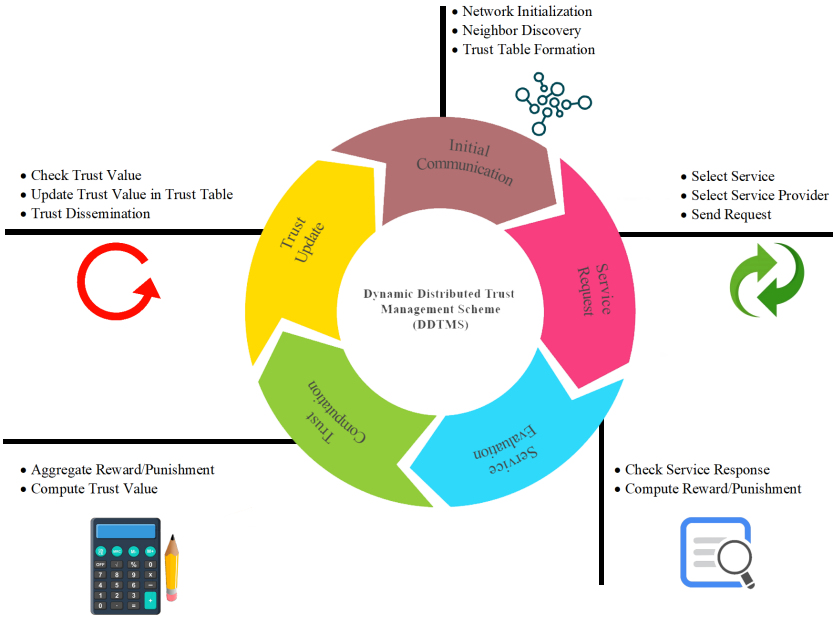

Figure 1: Process flow diagram of the proposed dynamic distributed trust management scheme (DDTMS).

list, and trust value. The trust table of each node is formulated in this phase. The information of each neighbor which is in the same radio transmission range is filled in the trust table. At a particular time, each node either provides a service(s) to OR requests a service from the neighbor node(s). In the beginning, the initial trust value of each node is set to 0 assuming they are unknown initially. Then only by communication and interaction with peer nodes, trust value will be updated to establish their trust status for subsequent data exchange.

Phase 2: service request

In the distributed model, every node is autonomous in making the decision of service request, awarding a good node and punishing a misbehaving node. Since this is a multiservice context, therefore, various nodes in the network can provide different services. As mentioned in the first phase that the trust table contains information about the neighbor node(s) including the node id, trust value, and service list. In this phase, the requester node first decides and selects which service to request then it examines which of its neighbor node provides the same service. To avail a particular service, a node randomly selects any of its neighbor providing that service from its trust table. As soon as the requester node selects its provider node then it sends the service request to the same node.

Phase 3: service evaluation

In the service evaluation phase, service is evaluated based on the response value. Once the request is sent, the requester node waits for a response for a specified round-trip-time (RTT). This waiting time (RTT) is between 500 ms (0.5 s) and 1000 ms (1 s) on average. If the provider node provides service within specified response time, then it is considered a successful service and granted reward score. On the other hand, if the provider node does not provide service within time, then it is considered the failed request and is penalized with a penalty score. To counter long waiting times caused by network delays the RTT time for a malicious node is considered double of that of a cooperating node. After each transaction, a response value is assigned to the provider node by the

requester node, depending upon the success or failure of the transaction. Each service has dynamic reward and penalty score which is assigned to the provider node according to its behavior. In the proposed DDTMS model, following weight criteria is used for service evaluation as described in the Table 1.

Table 1: Weight system for service evaluation.

Caption	Parameter	Mathematical equation	Equation no.
Reward	Note value	$N_j = Rs * W_s$	(1)
	Reward score	$Rs = 1 * ST/TT$	(2)
Punishment	Note value	$N_j = Ps * W_s$	(3)
	Penalty score	$Ps = -Log_{10}(2 * (FT/TT + OnOff) * FR)$ $where$ $FR \neq 0$	(4)
Weight	Service weight	$W_s = S_j * \sigma$ $where$ $0 < \sigma < 1$	(5)
Trust computation	Node trust value	$T_{np} = \sum N_j$ $where$ $-1 < T_{np} < 1$	(6)

Where the note value N_j assigned to a provider node by a requester node is based on whether the requested service S_j is granted or rejected. Rs is the rewarded score, ST is successful transaction, FT is failed transaction and TT is total transactions. In order to counter accidental service rejection caused by network issues, the proposed DDTMS makes use of Failed Repeated (FR) value i.e. if the last transaction is failed then another consecutive failed transaction would be considered as failed repeated. On-off is a counter value i.e. if transaction switches from successful to failed transaction each time, then this value is incremented. The S_j is a value allocated to a particular service. The weight Ws is assigned to every service. An adjust factor σ is a value which fluctuates between 0 and 1. If its value is higher the trust value also converges rapidly.

In the network, each service is valued according to its relative importance and therefore its value can be fine tuned as per the application environment. For instance, the service (S1) hosted to nitrogen dioxide sensor node has higher value S1 = 0.15 in that context whereas the same service (S1) hosted on some other node has low value S1 = 0.05 in a different context.

Phase 4: trust computation

In the this phase, the trust value of each node is particularly computed based on its reward and punishment value(s). These values are awarded according to the node's behavior after every transaction. Therefore, with these values, the final trust value (T_{np})is computed by requester node p of the provider node n, using the weighted sum system. Final trust value of the node is outlined as equation (6) in Table 1. Where T_{np} is the final Trust value of a particular node. The trust value of every node ranges from –1 to 1. The lowest value of the trust (also known as distrust) is –1, which any node can attain in trust table while 1 is the highest score of trust i.e. maximum trust of a node.

Algorithm 1 Dynamic trust computation.

Require: TransactionsHistory
Ensure: nodeTrust
Initialization
$ST = 0,\ Rs = 0,\ \sigma = 1,\ Ws = Si * \sigma,\ TT = 1$
$nodeTrust = 0,\ FT = 0,\ FR = 0$
Let Node s(Requester) requests Node t(Provider) for Service Si
Evaluate the Response from Requester
if $response == true$ **then** Calculate Reward
procedure CALCULATEREWARD
$ST = ST + 1$
$Rs = 1 * ST/TT$
$Reward(Nj) = Rs * Ws$
$lastInteraction = true$
end procedure
else
Calculate Punishment
procedure CALCULATEPUNISHMENT
$FT = FT + 1$
$TT = FT + ST$
$OnOff = OnOff + 1$
if $lastInteraction == false$ **then**
$FR = FR + 1$
end if
if $FR <> 0$ **then**
$Ps = -Log_{10}((2 * (FT/TT + OnOff) * FR)$
else
$Ps = -Log_{10}((2 * (FT/TT + OnOff))$
end if
$Punishment(Nj) = Ps * Ws$
$lastInteraction = false$
end procedure
end if
Calculate Trust
procedure CALCULATETRUST
$nodeTrust = Reward(Nj) + Punishment(Nj)$
if $nodeTrust < 0 AND nodeTrust > -0.33$ **then**
$nodeTrustCategory = low$
else
$Ps = -Log_{10}((2 * (FT/TT + OnOff))$
end if
end procedure
Return $nodeTrust$

In the proposed model, a node's trust category is attributed low if its trust value falls in between 0 and –0.33; mild if in range –0.34 to –0.66 and severe if from –0.67 to –1. Sometimes, requested services may be rejected due to some intermittent network issues therefore, a node may not be penalized immediately rather giving the flexibility to regain its trust and falls in low category to observe further behavior. In case the selected node further rejects more requested services then it lies in the mild category and needs several successful transactions to achieve positive trust among peer nodes in the network, unlike the low category. However, if

such a node provides some of the requested services but also continues to reject services alternately then it must be penalized for such selfish behavior. Repeated service rejection can result in the severe category and nodes stop communicating with such nodes.

Whenever a requester node n requests any service S_j from a provider node p, and provider node p successfully offers requested service to the requester, then accordingly it is rewarded by requester node n. This reward results in an increase in the provider's trust value depending on requested service i.e. S1, S2, or S3. The provider node p trust value increases as per the aforementioned system of weights. Similarly, any misbehaving node that does not offer the services requested, are punished by the requester node n accordingly. In this case, the trust score of the node reduces every time it does not offer the requested service. Therefore, due to repeated failure of providing requested services, trust value highly decreases. Such behavior of the node is considered as malevolent and is penalized with a dynamic penalty score. In the proposed model, the suspicious behavior is articulated in negative value; a trust score closer to –1 denotes a higher distrust of the node while the trust value closer to 1, depicts the node is more trustworthy for providing the announced services.

Algorithm 2 Identify and disseminate malicious node(s) information.

Require: NeighborList
Ensure: Disseminate Malicious Node Information
Let NeighborList is the list of neighbor nodes of Node_i
if $nodeTrust < 0$ and $nodeTrust \geq -0.33$ **then**
$nodeTrustCategory = low$
else if $nodeTrust < -0.33$ and $nodeTrust \geq -0.67$ **then**
$nodeTrustCategory = mild$
else if $nodeTrust < -0.67$ **then**
$nodeTrustCategory = severe$
end if
procedure CreateMaliciousList($NeighborList$)
$createMaliciousList$
for $i \leftarrow 1$ to $NeighborList(Size)$ **do**
$Node_\text{i} \leftarrow Neighbours$
if $Node_\text{i}.nodeTrustCategory == severe$ **then**
$addtoMaliciousList$
end if
end for
end procedure
procedure DisseminateMaliciousNodeInformation($TrustedList$, $MaliciousList$)
for $i \leftarrow 1$ to $TrustedList(Size)$ **do**
$addTrustedList_\text{i}$ to $MulticastAddressList$
end for
$SendMaliciousList$ to $MulticastAddressList$
end procedure

Phase 5: trust update

Once the trust value of a node is computed in the aforementioned phase, then the next job is to update this trust value in the trust table of the requester node. The newest trust value of the node is available in the trust table on completion of each transaction, irrespective of a negative or a positive value. After the trust value update in the trust table, if the latest trust value is positive score (near to 1) then node is considered as

cooperating node. On the other hand, if the updated trust value falls in mild category, then the requester node suspends all its future interactions with that particular service provider node. Furthermore, if the trust value of a node significantly downgrades and gets to the severe category, then requester node not only suspends its future correspondence with it but also shares such information with its peer nodes (except the identified malicious node). In case a node receives malicious behavior notification from peer node, it increments the failed repeated (FR) value for that particular node. The aforementioned procedure of trust computation is summarized in Algorithm 1. After identifying malicious behavior of a certain node, the requester node shares malicious node's status with its neighbor nodes (except the identified malicious node) using Algorithm 2.

4. Experiments, simulation, and reaction

To evaluate the performance of our proposed model, a net-based simulation tool has been developed and the main dashboard of this simulation tool is illustrated in Figure 2.

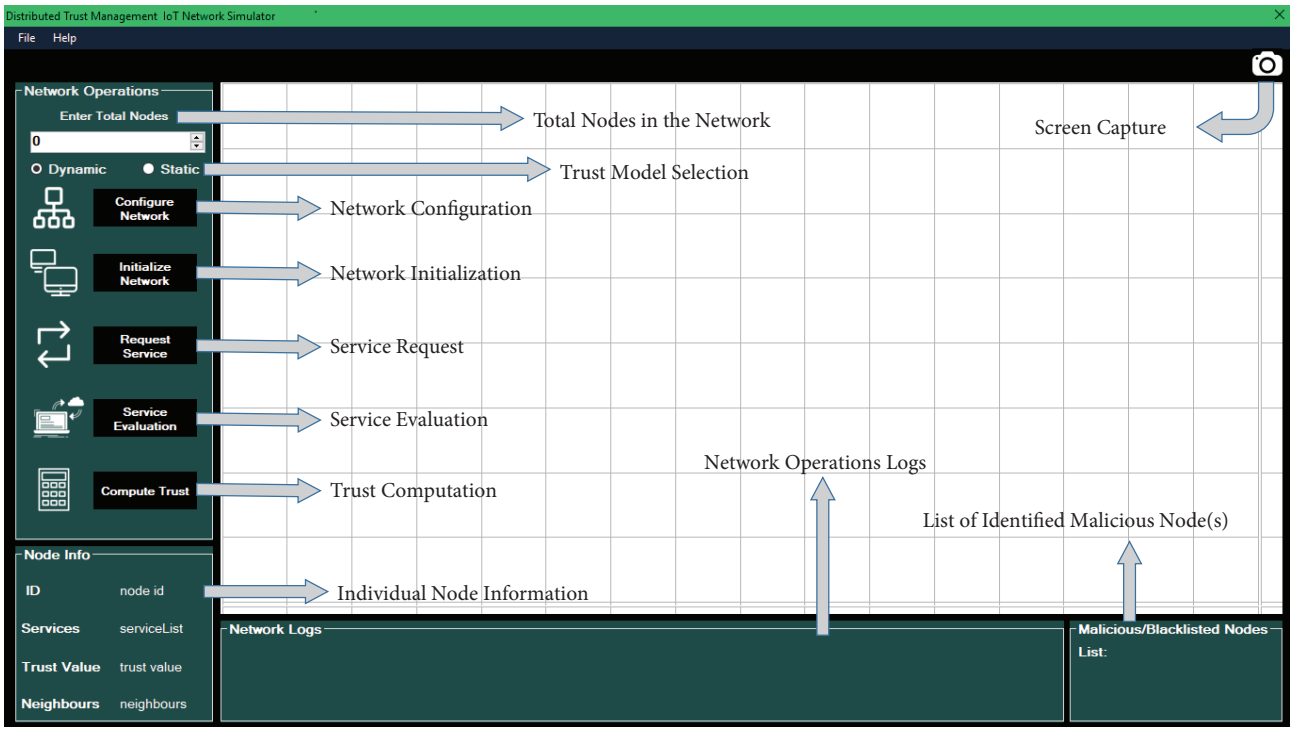

Figure 2: Main dashboard of the distributed trust management simulator for IoT.

This tool is useful to form a distributed network, randomly send a service request, evaluate service and calculate dynamic trust value based on the proposed algorithm. An integrated log facility provides all transactions statistics during simulation. Following two scenarios are implemented for on-off attack (OOA). Malicious node(s) provides and stops requested services alternately in on-off attack, exhibiting inconsistent behavior. This behavior needs to be investigated and identified so that nodes refrain from interacting with such nodes in future.

4.1. Scenario-I

To analyze the proposed algorithm, an IoT network is created consisting of 15 nodes. All nodes are deployed randomly across the network as shown in Figure 3.

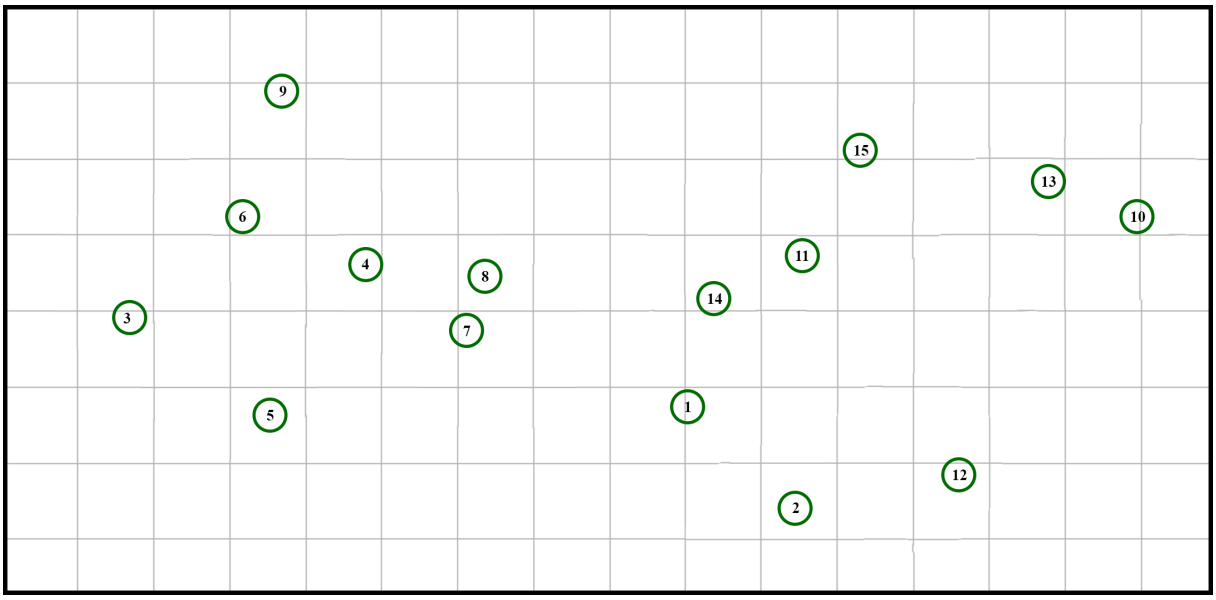

Figure 3: Nodes after deployment in scenario-I.

Moreover, in this network initialization stage, trust table of each node is also created which includes information like neighbor node id, service list, and corresponding trust values.

Considering trust value range between –1 and 1, initial trust value of a node is set to 0 (neither malicious nor trustworthy). Every node sends a random service request to any of its neighbor node from its trust table. The request interval is specified in the preferences. The deployed nodes in the network offer multiple services namely S1, S2, and S3 with 0.1, 0.05, and 0.02 values respectively. In the simulation, the execution of on-off attack by certain node(s) is demonstrated with the help of following scenario.

As shown in Figure 4, there are five nodes i.e. 5, 6, 7, 8, and 9 in neighborhood of node 4. In this case, node 4 requests different services from its neighbors in a random manner. Network log shows request time, request id, requester node id, and provider node id. After every service request, the requester node waits for specified round trip time (RTT) for service response by the provider. Based on failed or successful transaction, punishment or reward score is granted to the provider node in trust table. All neighbor nodes except node 8 offer requested services to node 4. Node 8 did not provide service requests most of the times. Due to this higher rejection, it is punished and gained a higher penalty score during the evaluation of service. Such of its behavior is considered as inconsistent. After getting a response from requester node(s), service reward or penalty score is calculated, then, service reward and punishment value (Nj) is evaluated based on reward or penalty score during service evaluation. Similarly, service evaluation is done for all nodes in the network.

Finally, the trust value of all nodes is computed based on their reward and punishment values. As the node 1 and node 8 has a higher penalty score, therefore, attained negative trust value. Simulation tool detected node 1 and node 8 as malicious nodes and marked red to depict malicious node in the network. Figure 5 shows the detected malicious node in the network.

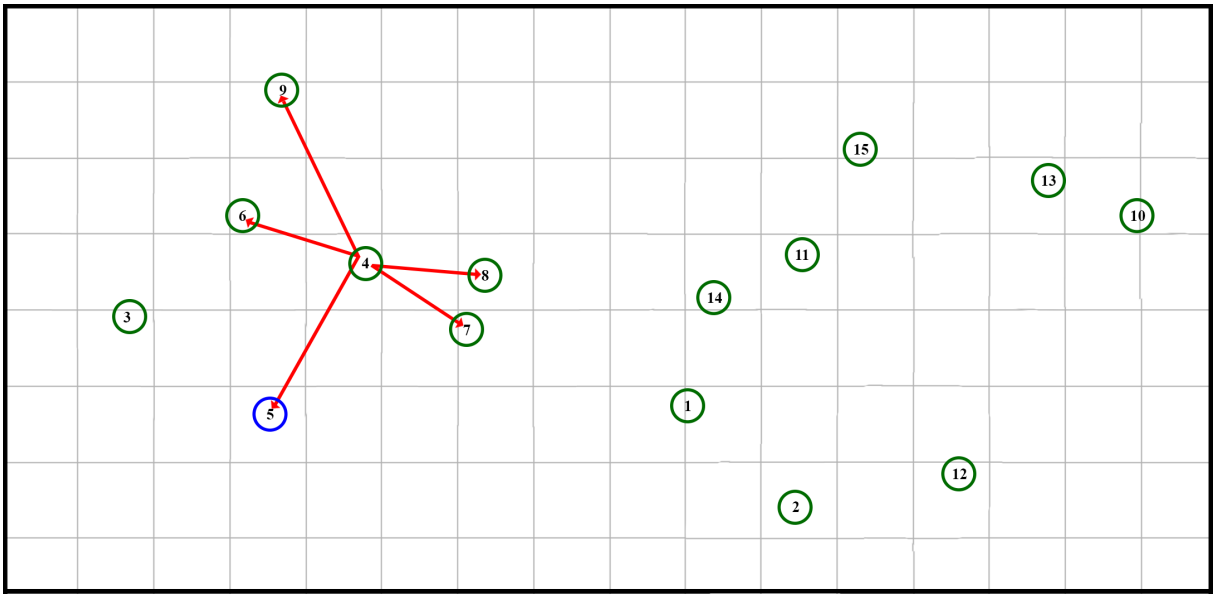

Figure 4: Demonstration of service request by node 4 to its neighbor nodes.

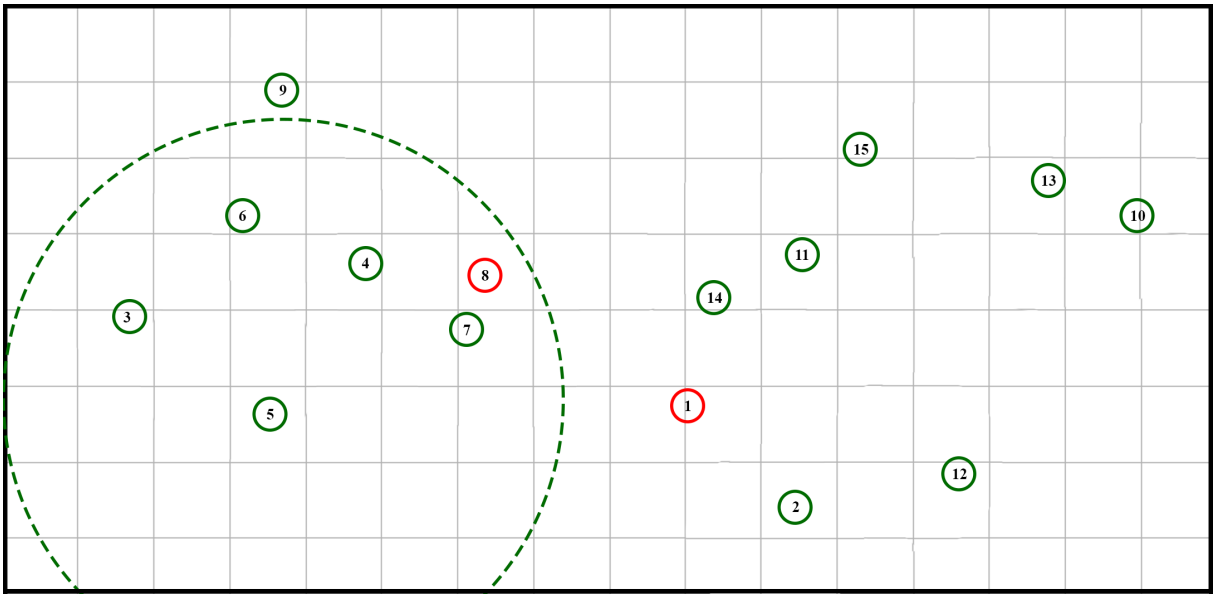

Figure 5: Trust computation and demonstration of malicious nodes (highlighted) detection using DDTMS.

The graph of identified malicious nodes i.e. node 1 and node 8 is shown in Figure 6. It depicts the negative trust values of both nodes gained after a various number of transactions. It can be seen that both nodes penalized with maximum distrust values (–1) in a short time. Node 1 gained distrust score after 10 transactions while node 8 gained its maximum distrust value i.e. –0.85 after 13 transactions.

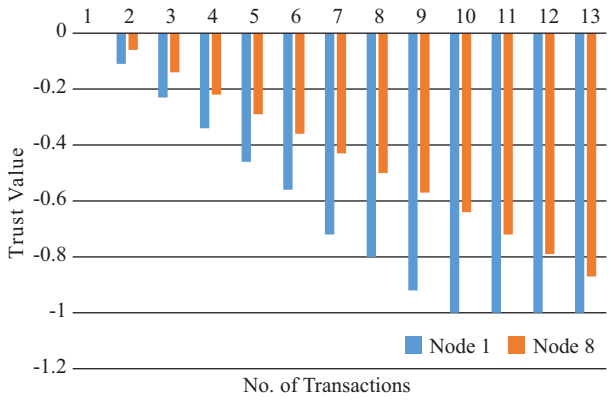

Figure 6: Trust values of malicious nodes 1 and 8 after different transactions.

4.2. Scenario-II

In the second scenario, another network consisting of 25 randomly deployed nodes is created as shown in Figure 7. Node 8 has total of 9 neighbors i.e. node 6, 10, 11, 12, 15, 17, 21, 23, and 24 as shown in Figure 8. The selected node 8 is sending a random request to all of its neighbors one after another.

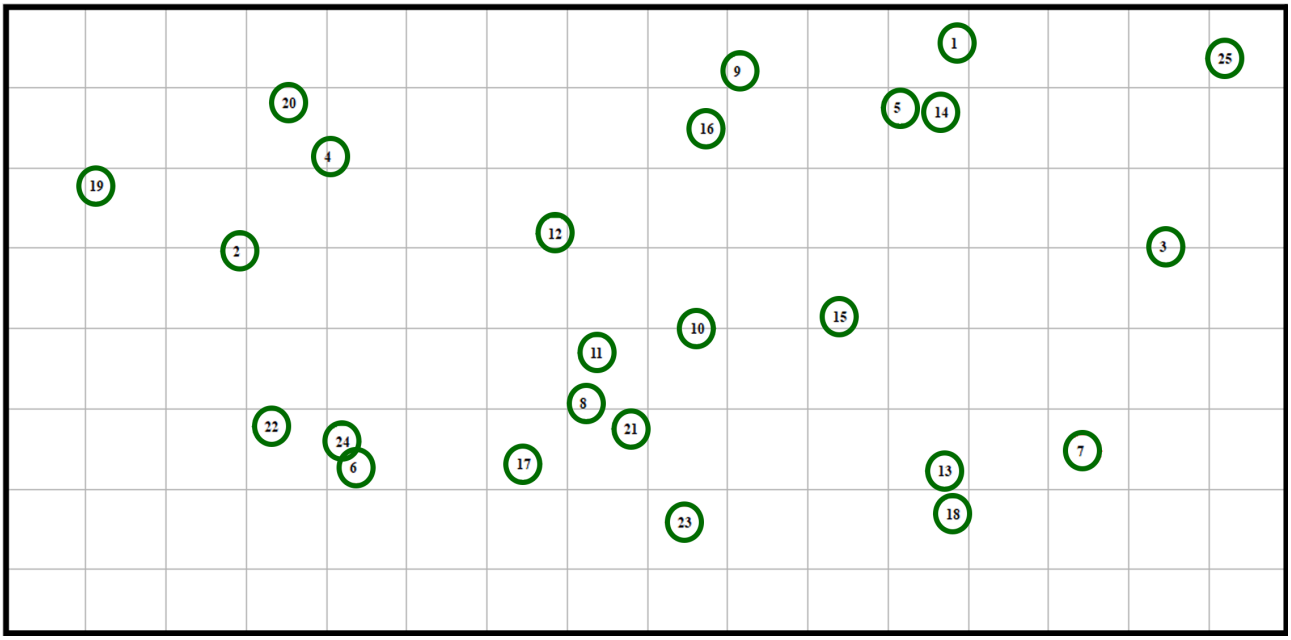

Figure 7: Nodes after deployment in scenario-II.

After every service request, node 8 waits for specified round trip time (RTT) for service response by the provider. It assigns a response value to each requested node for the type of service accordingly. Total 6 nodes out of 25 nodes identified as malicious in this scenario. The selected node 8 has detected 2 of its neighbors as malicious nodes while 7 honest nodes as shown in Figure 9. The trust value of each neighbor node is updated in the trust table of node 8. Similarly, other nodes also update the trust values of their neighbors after the trust score is computed. All malicious nodes are marked red and listed in the malicious node list. In the graph of all 6 malicious nodes, the assigned trust values by the simulation tool can be seen in Figure 10. Malevolent nodes

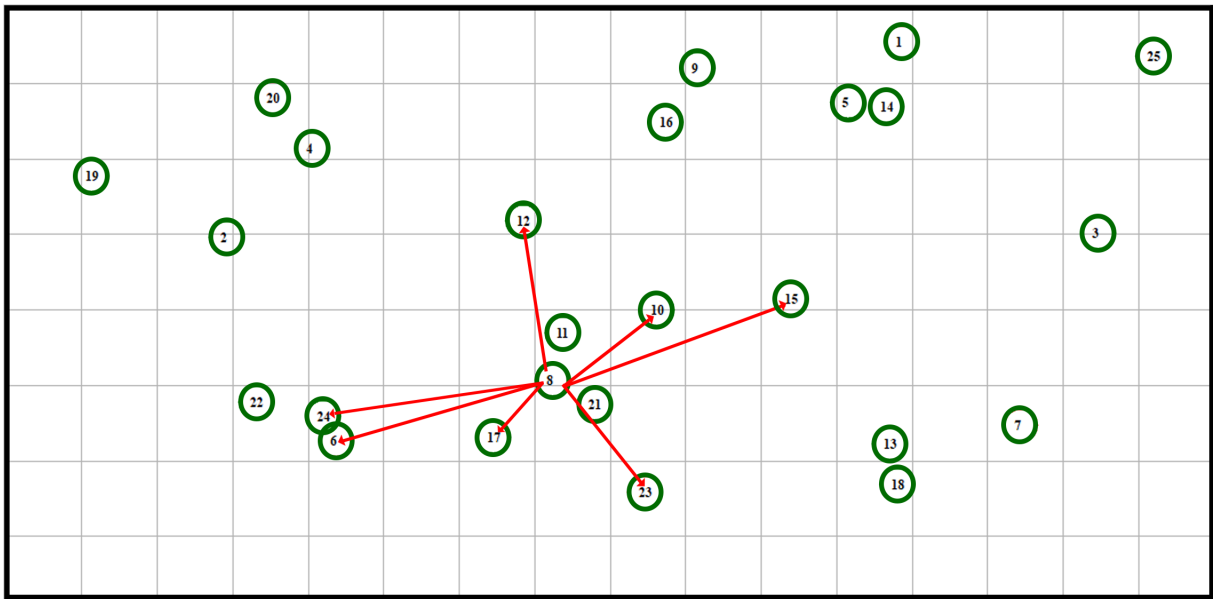

Figure 8: Demonstration of service request by node 8 to its neighbor nodes.

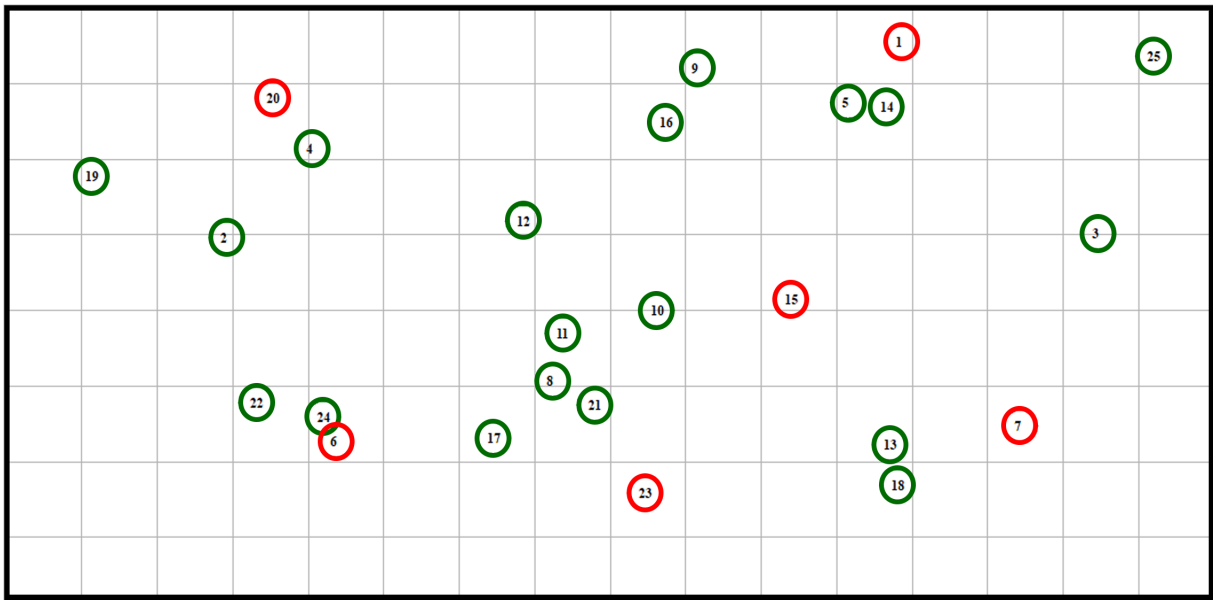

Figure 9: Demonstration of malicious nodes detection (highlighted) using DDTMS.

are penalized with maximum distrust values according to their behavior in various transactions.

Finally, a network comprising of 50 randomly deployed nodes is created and similar operations performed as in the aforementioned 2 scenarios. In this case, a total of 14 nodes out of 50 nodes are identified as malicious nodes as seen in the Figure 11. Node 1, 2, 5, 9, 10, 24, 25, 26, 32, 34, 35, 38, and 39 are identified as malevolent nodes.

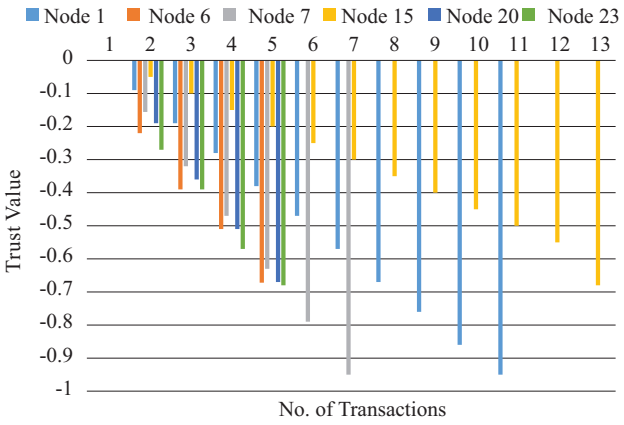

Figure 10: Trust values of identified multiple malicious nodes after different transactions.

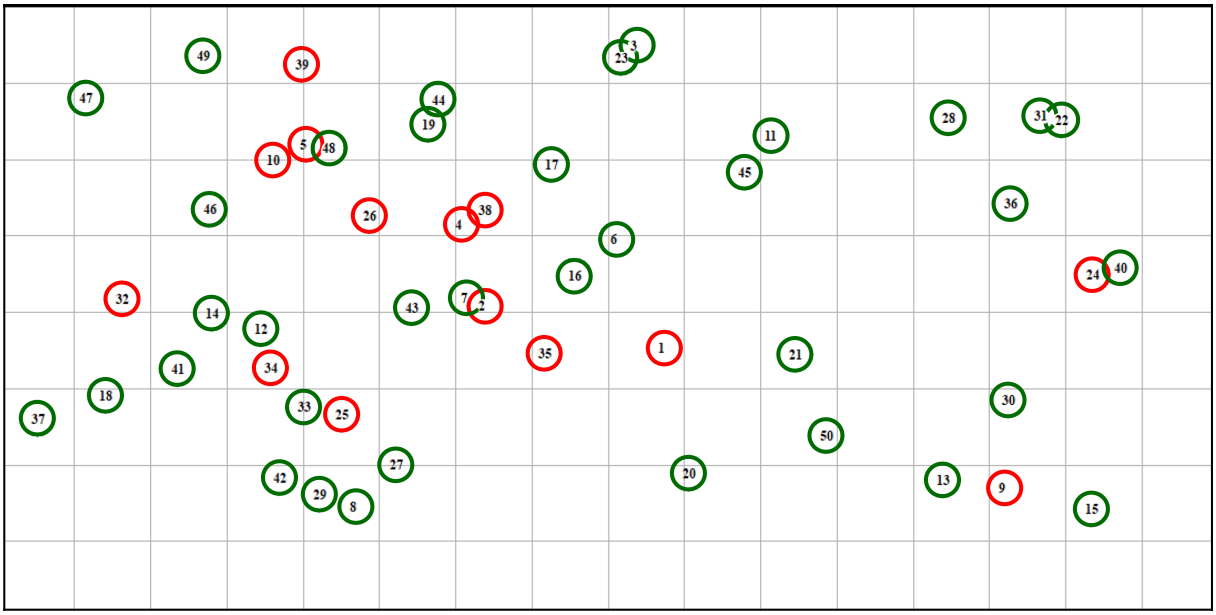

Figure 11: Demonstration of 14 malicious nodes detection (highlighted) out of 50 deployed nodes using DDTMS.

4.3. Reactions against attacks

In order to demonstrate the effectiveness of proposed Dynamic Distributed Trust Management scheme; its conduct towards On-Off attack is evaluated, that can occur in trust management systems.

4.3.1. On-off attack (OOA) protection

Simulation results in the aforementioned executed scenarios explicitly demonstrate the protection against on-off attacks by the proposed dynamic model. In the first scenario, 2 nodes are detected as malicious out of 15 nodes while 6 nodes out of 25 nodes in the second scenario and likewise 14 nodes out of 50 nodes in final case are identified as malicious nodes. Once the malicious node(s) are identified, then a multicast message is sent to all other cooperating neighbors to notify about the misbehaving node. Algorithm 2 is implemented in

the simulation tool to identify and disseminate malicious nodes information using multicast mechanism. The behavior of the model against on-off attack remains the same for different scenarios.

5. Performance analysis and validation

The performance of the proposed model can be examined by looking into various factors i.e. trust growth, trust decrease, and trust growth after decrease. Finally, the proposed model performance is compared with an existing relevant scheme from literature to evaluate its effectiveness and efficiency.

5.1. Dynamic trust adaptation

A node is awarded Reward as long as it successfully provides requested services to neighbor node(s). Higher reward score results in an increase in the trust value. The trust growth depends upon the frequency of interactions with peer nodes and may take a bit long time in gaining the maximum trust. Moreover, the trust value of a node dropped in case of on-off attack where malicious node selectively rejects offering services to neighbor node(s). However, a node may not be considered malicious immediately on account of rejection of service request once or twice but rather in such case, its trust value will be decreased. Furthermore, in case of consistent rejection of requested service, its trust value may finally be dropped to maximum distrust value ranging from –0.67 to –1 and classifies as severe category. A node may regain its trust if it starts cooperation in providing services to the peer nodes. This is applicable only in the case, when its trust category has not yet reached to severe. However, when a node is declared as of severe trust category, then the requesting node will give up all its future interactions with such malicious nodes. Figure 12 shows trust growth after decrease for three different nodes having low, mild and severe trust categories respectively. Even the low and mild category nodes, will go through large number of interactions to achieve positive trust value. This is because, nodes declared as low or mild category are awarded lower scores once they start cooperating again with the peer nodes.

Figure 12: Dynamic trust adaption in terms of low, mild, and severe categories.

5.2. Trust comparison

In the trust comparison, three cases are discussed, in which the proposed trust management scheme is compared with an existing trust management scheme [14]. In the first case, random patterns of successful and failed transactions are followed during simulation time using both trust management schemes. Figure 13 shows trust values of two malicious nodes at different instants of time during simulation. As benchmarks, we count the

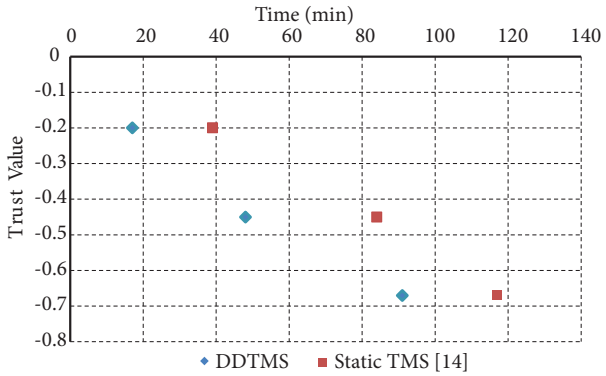

Figure 13: Trust comparison for random pattern of failed/successful transactions.

number of evolved transactions when the trust value reaches to –0.2, –0.45, and –0.67 (low, mild, and severe category, respectively) since simulation begins. Using the proposed DDTMS, it only requires 17, 45, and 91 transactions to acquire trust score of -0.2, -0.45, and -0.67 whereas the existing trust management scheme [14] reaches to similar trust values at transaction number 39, 69, and 117. Thus, the proposed DDTMS requires considerably less time in identifying malicious behavior of nodes compared to existing scheme.

The second case describes the comparison of both models considering a fixed pattern of successful and failed transactions to analyze the behavior of a node. In this case, the first 3 transactions are failed and the next 7 transactions are successful and this pattern of failed vs. successful (3:7) transactions is followed for the entire 75 transactions. The distribution of successful and failed interactions between a pair of nodes is given in Table 2. On top of the transaction's row, the total number of transactions is shown. Following total transactions, the left column indicates successful (S) while the right column represents failed (F) transactions out of total transactions in each column. The proposed model rewards lower trust scores for positive interactions while penalizes more for repeated failed transactions. Nevertheless, due to a higher number of successful transactions (7), the static trust model rewards more positive trust values while penalized less even after repeated failed transactions (3) and failed to detect such a malicious behavior. Over the course of 75 transactions, the proposed model concludes the behavior of the node as malicious thereby assigning –0.86 trust value whereas the static model is unable to detect malevolent behavior thereby awarding a positive trust value of 0.3 as illustrated in Figure 14.

Table 2: Comparison of trust values after various successful/failed transactions-I.

Transactions	10		20		30		40		50		60		70		75	
	S	F	S	F	S	F	S	F	S	F	S	F	S	F	S	F
	7	3	14	06	21	09	28	12	35	15	42	18	49	21	51	24
Static trust	0.10		0.20		0.30		0.40		0.50		0.60		0.70		0.30	
Dynamic trust	0.25		0.39		0.40		0.32		0.18		–0.01		–0.24		–0.86	

In the third case, the first 2 transactions are successful while 3rd transaction is a failed one (2:1) as shown in Table 3, unlike the first case. The static trust model rewards positive trust value one for each of 2 successful transactions and penalized twice negative value for the third failed transaction thereby resulting

Table 3: Comparison of trust values after various successful/failed transactions-II.

Transactions	10		20		30		40		50		60		70		75	
	S	F	S	F	S	F	S	F	S	F	S	F	S	F	S	F
	7	3	14	06	20	10	27	13	34	16	40	20	47	23	50	25
Static trust	0.05		0.10		0.00		0.05		0.10		0.00		0.05		0.00	
Dynamic trust	0.17		0.18		–0.04		–0.17		–0.32		–0.68		–0.68		–0.68	

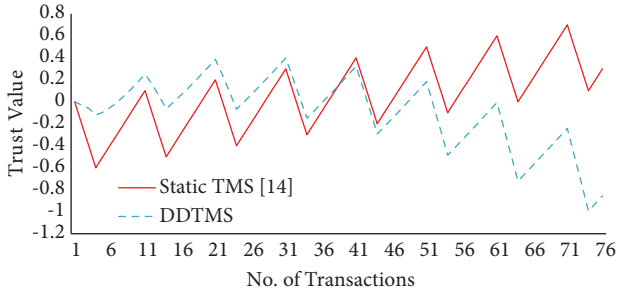

Figure 14: Trust comparison in case of fixed pattern i.e. 3:7 failed and successful transactions.

in 0 (same as the initial trust value) for this pattern. This behavior continues for the entire 75 transactions, which indicates the failure of the static model for this scenario in identifying malicious behavior. However, the proposed DDTMS successfully evaluates the aforesaid inconsistent pattern of the transactions by assigning negative trust values i.e. –0.17, –0.32, and –0.68 accordingly due to repeated failed transactions. Once the trust value reaches the severe category then it stops further communication as the –0.68 trust value remains the same for the remaining 15 transactions, as depicted in Figure 15.

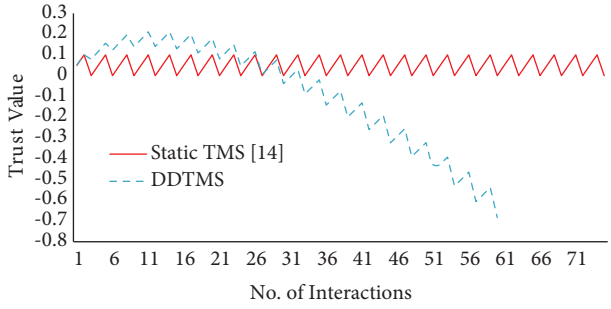

Figure 15: Trust comparison in case of fixed pattern i.e. 2:1 successful and failed transactions.

6. Conclusion

In this paper, a dynamic distributed trust management scheme (DDTMS) is proposed with major focus on identifying and mitigating on-off attacks (OOA) in the IoT. The proposed model evaluates the node's behavior with regards to providing or rejecting the requested services thus computing dynamic trust value of peer nodes in the network. DDTMS dynamically adjusts the reward scores depending upon to which category the node's trust level belongs to thereby making it difficult for misbehaving nodes to easily change their status from negative to positive. Moreover, once a one is identified as malicious on account of reaching to severe category, the requesting

node not only stops its future interactions with it but also informs other neighboring nodes. Demonstrated by simulation results, the proposed dynamic trust model successfully identifies the on-off attack of a malicious node in significantly less time thereby requiring fewer interactions compared to existing trust management scheme. Part of the future work, we intend to incorporate measures for countering other attacks like ballot-stuffing, and bad-mouthing in our methodology. Moreover, in addition to direct observation, we plan to include reputations from peer nodes in dynamically computing the trust value which is vital in countering various types of attacks in IoT environment.

References

[1] Curado M, Madeira H, Da Cunha PR, Cabral B, Abreu DP et al. Internet of Things. In: Kott A (editor). Cyber Resilience of Systems and Networks. 1st ed. New York, NY, USA: Springer International Publishing, 2019, pp. 381-401. doi: 10.1007/978-3-319-77492-3_16

[2] Dabbagh M, Rayes A. Internet of Things security and privacy. In: Internet of Things from Hype to Reality. 2nd ed. New York, NY, USA: Springer International Publishing, 2019, pp. 211-238. doi: 10.1007/978-3-319-99516-8

[3] Awan KA, Din IU, Almogren A, Guizani M, Khan S. StabTrust—a stable and centralized trust-based clustering mechanism for IoT enabled vehicular ad-hoc networks. IEEE Access 2020; 8: 21159-21177.

[4] Neeraj AS. Internet of Things and trust management in IoT–review. International Research Journal of Engineering and Technology 2016; 3 (6): 761-767.

[5] Issarny V, Bouloukakis G, Georgantas N, Billet B. Revisiting service-oriented architecture for the IoT: a middleware perspective. In: International Conference on Service-Oriented Computing (ICSOC); Banff, AB, Canada; 2016. p. 3-17.

[6] Xu H, Yu W, Griffith D, Golmie N. A survey on industrial Internet of Things: a cyber-physical systems perspective. IEEE Access 2018; 6: 78238-78259.

[7] Conti M, Dehghantanha A, Franke K, Watson S. Internet of Things security and forensics: Challenges and opportunities. Future Generation Computer Systems 2018; 78 (2): 544-546. doi: 10.1016/j.future.2017.07.060

[8] Jabeen F, Hamid Z, Akhunzada A, Abdul W, Ghouzali S. Trust and reputation management in healthcare systems: taxonomy, requirements and open issues. IEEE Access 2018; 6: 17246-17263.

[9] Firoozi F, Zadorozhny VI, Li FY. Subjective logic-based in-network data processing for trust management in collocated and distributed wireless sensor networks. IEEE Sensors Journal 2018; 18 (15): 6446-60.

[10] Ahmed A, Bakar KA, Channa MI, Haseeb K, Khan AW. TERP: a trust and energy aware routing protocol for wireless sensor network. IEEE Sensors Journal 2015; 15 (12): 6962-6972.

[11] Bao F, Chen R. Trust management for the Internet of Things and its application to service composition. In: IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM); San Francisco, CA, USA; 2012. pp. 1-6.

[12] Xie Y. A study on trust management algorithms for the social internet of things. Masters thesis, Nanyang Technological University, Singapore, 2016.

[13] Saied YB, Olivereau A, Zeghlache D, Laurent M. Trust management system design for the Internet of Things: a context-aware and multi-service approach. Computers & Security 2013; 39: 351-65.

[14] Mendoza CV, Kleinschmidt JH. Mitigating On-Off attacks in the Internet of Things using a distributed trust management scheme. International Journal of Distributed Sensor Networks 2015; 11 (11): 859731.

[15] Michiardi P, Molva R. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In: Blažič BJ, Klobučar T (editors). Advanced Communications and Multimedia Security. Boston, MA, USA: Springer, 2002, pp. 107-121.

[16] Ganeriwal S, Balzano LK, Srivastava MB. Reputation-based framework for high integrity sensor networks. ACM Transactions on Sensor Networks (TOSN) 2008; 4 (3): 1-37.

[17] Chen H, Wu H, Zhou X, Gao C. Agent-based trust model in wireless sensor networks. In: Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD); Qingdao, China; 2007. pp. 119-124.

[18] Reddy YB, Selmic R. Secure packet transfer in wireless sensor networks–a trust-based approach. In: IARIA, The Tenth International Conference on Networks (ICN); St. Maarten, Netherlands; 2011. pp. 1-6.

[19] Kim TK, Seo HS. A trust model using fuzzy logic in wireless sensor network. World Academy of Science, Engineering and Technology 2008; 42 (6): 63-6.

[20] Chae Y, DiPippo LC, Sun YL. Trust management for defending On-Off attacks. IEEE Transactions on Parallel and Distributed Systems 2014; 26 (4): 1178-1191.

[21] Mendoza CV, Kleinschmidt JH. A distributed trust management mechanism for the Internet of Things using a multi-service approach. Wireless Personal Communications 2018; 103 (3): 2501-2513.

[22] Chakravartula RN, Lakshmi VN. Trust management-framework for IoT-based P2P objects. International Journal of Peer to Peer Networks (IJP2P) 2017; 8(2/3): 1-20.

[23] Sharma A, Pilli ES, Mazumdar AP, Govil MC. A framework to manage trust in Internet of Things. In: International Conference on Emerging Trends in Communication Technologies (ETCT); Dehradun, India; 2016. pp. 1-5.

[24] Bernabe JB, Ramos JL, Gomez AF. TACIoT: multidimensional trust-aware access control system for the Internet of Things. Soft Computing 2016; 20 (5): 1763-1779.

[25] Chen R, Guo J, Bao F. Trust management for SOA-based IoT and its application to service composition. IEEE Transactions on Services Computing 2014; 9 (3): 482-95.

[26] Ruan Y, Durresi A, Alfantoukh L. Trust management framework for Internet of Things. In: IEEE 30th International Conference on Advanced Information Networking and Applications (AINA); Crans-Montana, Switzerland; 2016. pp. 1013-1019.

[27] Amol R. Detection of On-Off attack based on predictability trust in wireless sensor network. International Journal of Advanced Computational Engineering and Networking 2016; 4 (12): 2320-2106.

[28] Abderrahim OB, Elhedhili MH, Saidane L. DTMS-IoT: A Dirichlet-based trust management system mitigating On-Off attacks and dishonest recommendations for the Internet of Things. In: IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA); Agadir, Morocco; 2016. pp. 1-8.

[29] Awan KA, Din IU, Almogren A, Guizani M, Altameem A et al Robusttrust–a pro-privacy robust distributed trust management mechanism for Internet of Things. IEEE Access 2019; 7: 62095-62106.

[30] Sahoo RR, Sarkar S, Ray S. Defense against On-Off attack in trust establishment scheme for wireless sensor network. In: 2nd International Conference on Signal Processing and Communication (ICSPC); Coimbatore, India; 2019. pp. 153-160.

[31] Abderrahim OB, Elhdhili MH, Saidane L. TMCoI-SIOT: a trust management system based on communities of interest for the social Internet of Things. In: 13th International Wireless Communications and Mobile Computing Conference (IWCMC); Valencia, Spain; 2017. pp. 747-752.