International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 04 | April - 2023 Impact Factor: 8.176 ISSN: 2582-3930

Trust Computation in WSN System

Pushkaraj Kulkarni, Shrushti Kumatkar, Shubham Nandgaonkar, Ms. Megha V Gupta
Computer Engineering, New Horizon Institute of Technology and Management,
Thane, India

***Abstract*- Wireless sensor networks (WSNs) are emerging as a valuable technique for extracting information from the surrounding environment by deploying a large number of small-sized sensor nodes in sensitive, unsupervised, and (often) hostile environments. Traditional cryptographic methods are commonly used in WSN security. However, due to unattended and insecure deployment, a sensor node may be physically captured by an adversary who may obtain the underlying secret keys, or a subset thereof, in order to access the important data and/or other nodes in the network. Furthermore, a node might not function correctly due to a lack of resources or issues with the network link. In recent years, WSNs have been used to monitor the shifting behaviors of nodes in a network by using the fundamental concepts of trust and reputation. In light of these circumstances, we provide a Trust Computation in WSN System in this work. a Java-based trust and reputation models simulator designed to make it simple to test and evaluate trust and reputation models over WSNs. It allows the user to change numerous parameters, like the percentage of malicious nodes and the possibility of creating a collusion, among many others.**

Keywords—Trust & reputation management · Wireless sensor networks · Bio-inspired algorithms

I. Introduction

WSNs are networks built on the cooperation of little nodes. These nodes stand out for their low cost, low energy usage, and, of course, wireless communication. They can be used to measure temperature, pressure, humidity, lightness, etc., however at the moment they frequently have certain failure probability as well as severe computing, memory, and energy limitations. In many applications, including traffic control in Vehicle-to-Vehicle (V2V) networks, weather measurement, and fire detection, Wireless Sensor Networks (WSNs) are frequently used. WSNs typically contain a lot of these nodes, which, along with their extremely dynamic architecture, may cause some scalability issues.Since these networks have a variety of intriguing uses, including ITS (Intelligent Transportation Systems), sanitary applications, demotics, and the environment, many research groups are working on them. They frequently experience several security flaws, nonetheless, which leaves them frequently open to various assaults due to their significant limits. Hardware issues, for instance, can be the cause of incorrectly vital information spreading. But even worse, nodes that are a part of a WSN could act inappropriately when they are asked for a measurement or some data.

We will use a system where certain network nodes function as customers and request services, while others operate as servers or service providers, without losing the generality of the situation. In such a case, a node might respond to a request for a service by offering a false one. Additionally, because we presupposed one of the most restrictive scenarios in which each sensor can only communicate with its immediate neighbors (i.e., it cannot establish a direct connection with a node more than one hop away), a malicious node could avoid communicating with its helpful neighbors or always lead to other malicious nodes, forming a conspiracy. Thus, it is crucial to reliably discern reliable nodes from malicious ones. Through a trust and reputation mechanism, this reliable node identification can be accomplished. In this research, we explicitly describe a trust and reputation model for wireless sensor networks (WSNs) named BTRM-WSN (Bio-inspired Trust and Reputation Model for Wireless Sensor Networks) to carry out the selection of the most reliable node along the most reliable path providing a particular service. The ant colony system method serves as the foundation for our suggested model. We'll see later that this enables a client to communicate more frequently with a reliable server than with one that's acting inappropriately. As a result, we introduce Trust Computation in WSN System, a trust and reputation model simulator for WSNs that aims to provide a general tool for testing and comparing trust and reputation models. We included some of the most typical experiments found in the literature for this type of model, as well as an API that serves as a framework for simply incorporating new trust and reputation models into our simulator.

II . Background

In this section we will present a review of some of the most relevant and novel trust and reputation models over Wireless Sensor Networks.

I. BTRM-WSN

BTRM-WSN is a bio-inspired trust and reputation model for Wireless Sensor Networks that aims to find the most trustworthy path leading to the most respectable node in a WSN that provides a specific service. It is based on an ant colony system's bio-inspired algorithm, but because of the specific restrictions and limitations found in WSNs, the ACS cannot be directly used there. As a result, some changes must be made. In our approach, for example, each node keeps a pheromone trace for each of its neighbors. This pheromone trace will impact the likelihood of ants choosing one path over another, and can be interpreted as the degree of trust given by one node to another. BTRM-WSN is concerned with developing trust and reputation management mechanisms for WSNs that are inspired by biological systems, particularly the behavior of social insects such as ants and bees. The goal is to create a system in which individual sensors in the network can assess the trustworthiness of other sensors based on their past behavior and reputation, and use this information to make decisions about which sensors to rely on for data transmission and processing. In a WSN, sensors can use information about the reliability and trustworthiness of other sensors to make decisions about which sensors to communicate with and which data to trust. By using bio-inspired mechanisms, BTRM-WSN aims to improve the reliability and security of wireless sensor networks and make them more resilient to attacks and failures.

Algorithm 1: BTRM-WSN

1: while (condition) do
2: for k = 1 to Number_of_ants do
3: Sk ← initial sensor (client)
4: Launch ant k
5:
6: do
7: for every returned ant k do
8: if (Q(Sk) > Q(Current_Best)) then
9: Current_Best ← Sk
10: while (timeout does not expire) and
11: Num_returned_ants < %Number_of_ants
12:
13: if (Q(Current_Best) > Q(Global_Best)) then
14: Global_Best← Current_Best
15: Pheromone_global_updating
16: (Global_Best,Q(Global_Best),ρ)
17:
18: return Global_Best

II. EigenTrust

EigenTrust is a reputation management algorithm designed for use in peer-to-peer (P2P) networks. The algorithm is used to determine the reputation of individual nodes within the network based on their past behavior. The goal of the EigenTrust algorithm is to provide a way for nodes in the network to identify trustworthy peers, and to avoid interacting with nodes that are known to be unreliable. By doing so, the algorithm helps to improve the overall reliability and efficiency of the P2P network. The algorithm works by creating a trust matrix that represents the trustworthiness of each node in the network. This matrix is computed using a recursive formula that takes into account the opinions of other trusted nodes in the network. Each node maintains its own version of the trust matrix and updates it periodically based on the opinions of other nodes it has interacted with. The algorithm uses an iterative process to converge on a final set of trust scores for each node in the network. The final scores are based on a combination of the node's own behavior and the opinions of other trusted nodes in the network. Nodes that consistently exhibit trustworthy behavior are given higher trust scores, while nodes that exhibit untrustworthy behavior are given lower scores. One of the key benefits of the EigenTrust algorithm is that it is able to handle malicious nodes that try to manipulate the trust scores of other nodes in the network. This is accomplished by using a technique called "eigenvalue centrality" to identify nodes that are attempting to influence the trust scores of others. Nodes that are found to be engaging in this type of behavior are penalized and their trust scores are reduced. Overall, EigenTrust is an important algorithm for improving the security and reliability of P2P networks.

III. Peer Trust

PeerTrust is a trust management model used in peer-to-peer (P2P) networks. It is designed to improve the security and reliability of P2P networks by enabling peers to assess the trustworthiness of other peers based on their past behavior and reputation. In a P2P network, there is no central authority that can manage the trust relationships between peers. Instead, PeerTrust relies on the principles of distributed trust management, where each peer in the network is responsible for assessing the trustworthiness of other peers. PeerTrust uses a combination of direct and indirect trust mechanisms to assess the trustworthiness of peers. Direct trust is based on the direct interactions between two peers, such as data exchanges or resource sharing. Indirect trust is based on the recommendations of other trusted peers in the network. PeerTrust also uses a reputation system to maintain the trustworthiness of peers over time. Each peer is assigned a reputation score based on their past behavior in the network. Peers with high reputation scores are considered more trustworthy and are given higher priority for resource sharing or data exchange. Overall, PeerTrust is an important model for improving the security and reliability of P2P networks.

IV. Power Trust

PowerTrust is a trust management model that is used in peer-to-peer (P2P) networks to improve the security and reliability of resource sharing. It is based on the principle of using a distributed reputation system to assess the trustworthiness of nodes in the network.

Here, each node maintains a local reputation score for every other node based on its behavior in the network. The local reputation scores are then combined to generate a global reputation score for each node using a weighted power

iteration algorithm. The power iteration algorithm distributes weights to each node based on their reputation scores and utilizes these weights to update the network's overall reputation scores. The weights are modified based on feedback from other nodes in the network, which helps to ensure that node trust connections are always up to date and correct. PowerTrust also employs a feedback method to assist nodes in determining the trustworthiness of other nodes. Nodes can send feedback on their behavior to other nodes, and this feedback is utilized to update node trust relationships. Feedback can take many forms, such as ratings or reviews, and can be used to assist nodes in making informed decisions on which nodes to trust. Overall, PowerTrust is an important trust management paradigm for increasing the security and reliability of peer-to-peer networks. It can aid in the prevention of attacks and ensuring that network resources are shared efficiently and fairly.

V. LFTM

LFTM stands for "Layered Feedback Trust Management". It is a trust management model that is used in distributed systems to improve security and reliability by enabling nodes to assess the trustworthiness of other nodes based on their behavior. In LFTM, a layered method is used to maintain the trust connections between nodes. Depending on how nodes behave, each layer symbolizes a different amount of trust, and nodes can move up or down the levels. For instance, a node with trustworthy behavior might advance to a higher layer, whereas a node with malicious behavior might descend to a lower layer. In order to assist nodes in determining whether other nodes are reliable, LFTM also employs feedback mechanisms. The trust connections between nodes are updated using the input that nodes might give to other nodes regarding their behavior. Feedback can come in many different forms, such as ratings or reviews, and it can be used to help nodes decide which nodes are worth trusting. The LFTM is a significant trust management methodology for enhancing distributed systems' security and dependability. It can help to avoid attacks and guarantee that resources are distributed effectively and fairly in the system by enabling nodes to judge the trustworthiness of other nodes based on their behavior.

VI. TRIP

TRIP stands for "Trust and Reputation in Peer-to-Peer file sharing systems". It is a trust management model that is used in peer-to-peer (P2P) file sharing systems to improve security and reliability by enabling nodes to assess the trustworthiness of other nodes based on their behavior. In TRIP, nodes in a P2P network keep track of each other's reputation scores based on how they behave in the network. The number of files shared, the number of successful downloads, and the number of unsuccessful downloads are some of the variables that influence reputation scores. Nodes with higher reputation scores are regarded as being more reliable and are given precedence when sharing resources or exchanging data. In addition, TRIP makes use of a feedback system to assist nodes in determining the reliability of other nodes. Nodes have the ability to give feedback to other nodes about their actions, and this feedback is utilized to update the node-to-node trust relationships. Feedback can be offered in a variety of ways, such as ratings or reviews, and it can be used to guide nodes in choosing which nodes to believe. TRIP is a crucial concept for managing trust that may be used to increase the security and dependability of P2P file sharing networks. It can aid in the prevention of attacks and ensure that resources are distributed effectively and fairly throughout the network by enabling nodes to judge the trustworthiness of other nodes based on their behavior.

III. Related Work

Numerous network simulators have been created to test novel communication protocols and evaluate their accuracy, resilience, and correctness. For instance, the authors of “A survey of peer-to-peer network simulators” provide a survey of P2P network simulators, outlining both their key benefits and key drawbacks. Although there are several of these network simulators, distributed network simulators for trust and reputation models are scarce. Yang et al. “Inside attacker detection in Hierarchical Wireless Sensor Networks” proposed an architecture which is able to track targets with random movement patterns with accuracy over a wide range of target speed. Arora et al. “Implicit Security Authentication Scheme in Wireless Sensor Networks” developed an experimental wireless sensor network for distributed intrusion detection but it is not autonomic.

IV. Implementation

Each trust and reputation model has unique traits and features of its own. However, most of them adhere to the same abstract schema or pattern regarding the sequence of events that must occur in a distributed system using a trust and/or reputation model in order to complete a transaction in its entirety. Therefore, creating a trust and reputation models interface that was as generic as feasible was one of the key goals of our work. First of all, we determined the four primary processes that must be completed in the majority of these models.

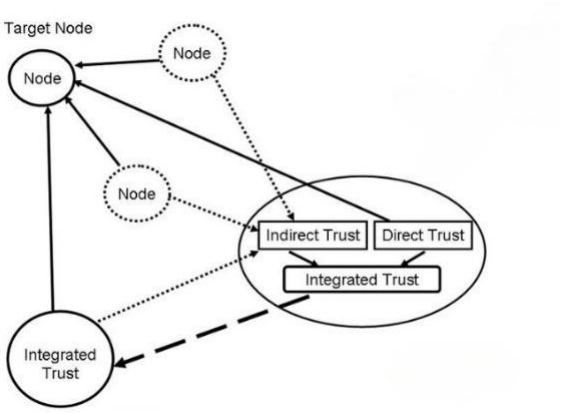

Fig 1. Proposed Architecture

We have developed an abstract Java class called TCModel_WSN containing one attribute: a set of generic parameters for trust and reputation models. To add a new trust and/or reputation model to the simulator, both TCModel_WSN and TCMParameters subclasses must be implemented. A subclass of the Service class could potentially be developed to specify additional details or attributes.

When using our simulator, the first thing you should do is create a new WSN. To do so, there are two areas where we

may specify the maximum and minimum number of sensors we want in our networks, as well as a slide bar to specify the wireless range of each sensor. These three characteristics will determine the network's link density.

Furthermore, we can decide how many of the nodes we want to function as customers needing a default service. Therefore, the remainder of them will serve as servers. We can also estimate the proportion of those servers that won't provide the required service and will instead operate merely as relay nodes. Last but not least, it is conceivable to estimate the proportion of servers that will be malevolent, that is, those that will not provide the service they are really offering but a poorer one or even no service.

Once we have customized all of those criteria to meet our needs, we can easily create a new random WSN by using the "New WSN" button at the bottom. By selecting the

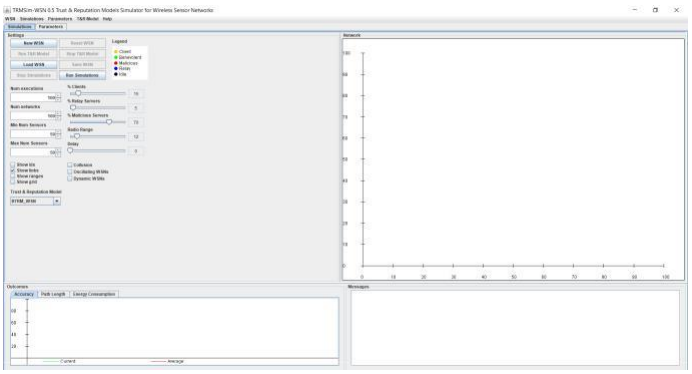

Fig 2. Homepage

"Load WSN" button, a WSN can also be loaded from an XML file, and the "Save WSN" button allows the current WSN to be saved into an XML file. Change the wireless range parameter and click "Reset WSN" to test the WSN we now have with a different links density.

The simulation settings need to be set up next. First, using the chosen trust and reputation model, we can calculate the number of simulation executions we need, or the number of times each network client will request its default service. According to the settings outlined in the preceding article, we may also choose the variety of random WSNs we desire. Regarding the visual or graphic depiction of the networks to be examined in our simulations, we can make certain decisions.

For instance, we can choose whether or not to display the wireless ranges, linkages between sensors, or unique identification for each sensor. With the help of the parameters panel, we can manually enter each parameter's value for the currently chosen trust and reputation model or configure the input parameters file. Since battery and energy consumption limitations are one of the key characteristics of WSNs, it is also possible to simulate a dynamic WSN in which some sensors switch to an idle state for a while if they do not get any requests within a predetermined amount of time. A sensor that is inactive neither receives nor sends any messages or packets.They awaken again after a predetermined timeout. Once we have established all the previous settings, we are ready to start our simulations. If we want to run a simulation only over the current network,

we should press the “Run WSN” button. By pressing the "Stop WSN" button, we may make the simulation end immediately.

Otherwise, we must use the "Run Simulations" button if what we want is to conduct a simulation over a specified number of random WSNs. When we click the "Stop Simulations" button, the current results of the simulation will be displayed.

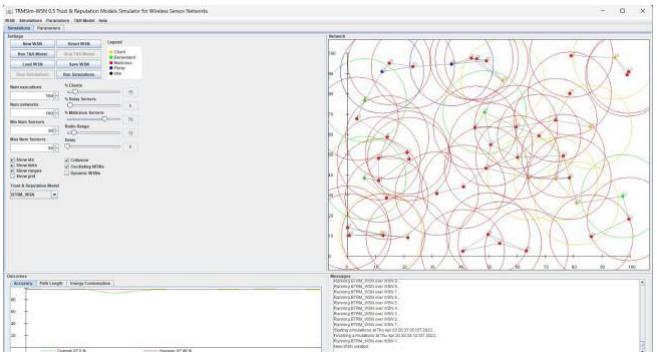

Fig 3. Trust and Reputation Models Simulator for BTRM-WSN

In the proposed system, each node is capable of calculating its own predictions for the best rated services. In addition, it computes the effectiveness of one node on another by combining their trust and similarity. However, the proposed recommender is not able to predict the rating for a device if somebody has not rated it. Moreover, it is difficult to tackle the prediction issues when searching nodes that are similar to each other.

V. Objectives of trust management

- Trust computation contributes to the identification and isolation of hostile nodes in the network, enhancing network security. Trust computation can assist in thwarting assaults like node replication, selective forwarding, and sinkhole attacks by determining the trustworthiness of nodes based on their behavior.
- By identifying trustworthy and dependable nodes, trust computation can aid in the effective allocation of resources. By doing this, it is possible to reduce resource waste and make sure that nodes are given the resources they will most likely need.
- By selecting nodes that are more likely to give accurate and trustworthy data, trust computation can help to increase the reliability of data. The quality and dependability of the data in the network can be increased through trust computation by giving greater trust scores to nodes that offer reliable data.
- By identifying and isolating nodes that are prone to failure or that have low reliability, trust computation can help to increase fault tolerance in WSN systems. Trust computation can assist in enhancing the network's overall availability and dependability by minimizing the employment of such nodes.

VI. Oscillating behavior and collusion

In order to test the accuracy of every simulated trust and reputation model we have included two security threats to our simulator. The first is related to the servers that are providing the requested service behaving oscillatory. If that option is chosen, each evil server will turn good after 20 executions (also known as transactions or interactions). Then, a random number generator selects the same proportion of previously harmful servers to be malicious now.

The second security risk is the potential for hostile servers to band together and commit collusion. This implies that every malicious sensor will rate every other malicious sensor at its maximum and every benign sensor at its lowest.A good trust and reputation model should quickly react against these behavioral changes and collusions and readapt itself in order to prevent selecting a malicious node as the most trustworthy or reputable one.

VII. Results

Based on the location information of sensor nodes, we observe three different kinds of mechanisms for choosing the recommender in this paper:

1) finding a recommender which is closest to the object node to save energy consumption,

2) finding a recommender which has the highest trust value to guarantee the reliability of Trust Chain and

3) finding an optimal Trust Chain by both considering. Two panels make it easier for us to understand what has occurred in the simulator so far or is happening right now, as well as the outcomes of the previous simulation. For instance, a number of messages are displayed in the messages panel and include essential information like the time the most recent simulation began or ended or the name of the WSN being tested at the moment.

Additionally, each action, like developing a New WSN, loading or saving the current one, or presenting ranges, IDs, and linkages, among other things, are also documented and exhibited in the IEEE ICC 2009 proceedings under the leadership of subject matter experts from the IEEE Communications Society. The results panel informs us of the outcomes of the active simulation network or the average results for the entire experiment.

Here, three significant values can be seen: the model's accuracy, the average length of every path found by each client in each simulated network, and the amount of energy the model uses (for future work).

It is simple to add new panels if more information about the tests has to be shown. By compiling client happiness data from each of the tested WSNs, the average satisfaction is calculated. Unfortunately, customers that are unable to connect to any helpful servers are not taken into account while calculating these results.

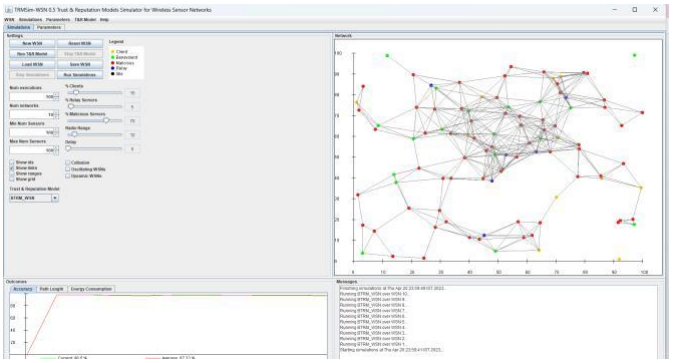

Fig 4. Trust and Reputation Models Simulator for BTRM-WSN showing links

In figure we can observe that a simulation over 10 random dynamic WSNs (with 100 sensors each one) has been carried out using BTRM-WSN model. There were a 15% of clients, an 8.5% (85% · 10%) of relay sensors, a 53.55% (85% · 90% · 70%) of malicious servers and a 22.95% (85% · 90% · 30%) of benevolent ones. The average number of hops needed to reach the most trustworthy server was 6.04 and the average percentage of times that the model selected a benevolent server as the most trustworthy one was 80%.

VIII. Conclusion And Future Work

Today, it is possible to test low level communication protocols using a variety of network simulators. To test the accuracy and validity of trust and reputation models for distributed systems, and specifically for WSNs, there aren't enough simulators available. In this study, we introduce TCM in WSN, a simulator of trust and reputation models for wireless sensor networks. We demonstrated the interface for our generic trust and reputation models and described how to quickly incorporate a new trust and reputation model into the simulator.

Also, we have covered TCM in WSN's features, potential applications, and configuration options for creating unique simulations. Nonetheless, TCM in WSN may use a number of changes and improvements. To determine the overhead caused by each simulated model, for example, we intend to add an energy consumption module. A further intriguing alternative would be the capability of choosing a particular sensor and altering its characteristics. We also intend to add a new feature to the TCM in WSN simulator that allows nodes in mobile wireless sensor networks to move and modify their positions over time.

Authors and Affiliations

[1] Y. Sun and Y. Yang, “Trust Establishment in Distributed Networks: Analysis and Modeling,” in Proceedings of the IEEE International Conference on Communications (IEEE ICC 2007), Communication and Information Systems Security Symposium, Glasgow, Scotland, jun 2007.

[2] A Survey of Trust Management for Wireless Sensor Networks" by R. Wang, B. Liang, and C. Liang, jan 2012

[3] S. Kamvar, M. Schlosser, and H. Garcia-Molina, “The EigenTrust Algorithm for Reputation Management in P2P Networks,” in Proc. of the International World Wide Web Conference (WWW), Budapest, Hungary, May 2003.

[4] K. Romer and F. Mattern, “The Design Space of WirelessSensor Network,” IEEE Wireless Communications, vol. 11, no. 6, pp. 54–61, dec 2004.

[5] "Trust-Based Routing in Wireless Sensor Networks: A Survey" by A. Khelil, M. Guerroumi, and T. Noel, May 2015.

[6] A. Boukerche, L. Xu, and K. El-Khatib, “Trust-based security for wireless ad hoc and sensor networks,” Computer Communications, vol. 30, no. 11-12, pp. 2413–2427, 2007.

[7] S. Naicken, A. Basu, B. Livingston, and S. Rohitbhai, “A survey of peer-to-peer network simulators,” Proceedings of the 7th Annual Postgraduate Symposium (PGNet ’06), 2006.

[8] B. L. Titzer, D. K. Lee, and J. Palsberg, “Avrora: scalable sensor network simulation with precise timing,” 2005, pp. 477–482.

[9] Y. Zhang, W. Wang, and S. Lu, “Simulating trust overlay in p2p ¨ networks,” in International Conference on Computational Science (1), 2007, pp. 632–639.