ISSN 2277-3061

Cooperative Intrusion Detection Technique in Wireless Sensor Networks

Nour El Din S. Eissa[1], Gamal I. Selim[2]
Department of Computer Engineering, Arab Academy for Science, Technology & Maritime Transport
[1]nsky90@hotmail.com
Department of Computer Engineering, Arab Academy for Science, Technology & Maritime Transport
[2]dgamal55@yahoo.com

ABSTRACT

Wireless Sensor Networks (WSN) are becoming more and more popular everyday due to their increasing ability to monitor certain phenomenon over wide regions such as air pollution, natural disaster, industrial monitoring and underwater applications [1]. The simple security capabilities of the nodes in the sensors network makes it an easy target for an intruder to take over some of the node(s) in the sensor network and to start altering the data received or sent by these nodes before forwarding it to the other nodes in effort to prevent the destination from properly decoding or reading the received data. These attacked nodes lead to tremendous amount of unusable data travelling across the network. The objective of this paper aims to detect these malicious nodes and cast them outside the network using a cooperative local voting approach with a dynamic center and study its effect on the amount of aggregated data though the network and the time required to deliver the sensors data.

Keywords

Wireless Sensor Networks (WSN), Network Coding, Malicious Node, Intrusion, Aggregated Data, WSN Delivery Time.

ISSN 2277-3061

I. INTRODUCTION

Wireless Sensor Networks consist of multiple nodes spatially connected together using an appropriate wireless technology such as Bluetooth, UWB [2] or IEEE 802.15.4 based-technologies depending on the size of the region that will be covered by the network and the sophistication of the devices used. Practically, due to the fact that most WSNs require large number of nodes, they are manufactured using low-cost components which imposes limited computational resources and energy to use. While sophisticated Sensor Networks implement techniques for energy harvesting [3] in effort to extend their functional lifetime, it is yet a challenge to create a complex security mechanism with such limited resources and power. As an estimate, each bit transmitted consumes as much power as executing 800-1000 instructions which renders mechanisms like public key cryptography very hard to implement. [4]

The lack of a rigid secure mechanism that defends the network from external intruders renders it an easy target to attack. The nodes in a sensor network usually send the data to the nearest fusion center which has the role of interpreting the received data by combining all the sensors readings together. After gathering enough information, the fusion center may be able to detect the presence of incoherent data and will start to back-cast messages to the network in order to detect the misbehaving nodes. The corrupted data might be the result of many external factors, such as channel noises, a physical damage to the sensor or it may be the result of a positive attack on the node. The fusion center is considered as a single point of failure in the network, thus, only relying on it in detecting the misbehaving nodes will be useless if it fails for any reason, and the data will have to travel further in the network to reach the next available fusion center. This will increase the amount of aggregated data exchanged between the fusion center and the area containing the affected nodes which is going to negatively impact the energy-efficiency of the nodes [5].

The paper researches the impact of the malicious nodes on the amount of aggregated data sent through the network and the resulting delay in delivering the overall sensors data to the destination and will provide a peer-based cooperative method to detect the malicious nodes without the need for a static communication fusion center. Network security attack techniques fall into two major categories, which are attack detection techniques and attack prevention techniques. The work in this paper aims on detecting the already compromised nodes but not preventing the attacks.

II. IMPACT OF NETWORK CODING ON WSN

In WSN, each node can act as client gathering the data about the phenomenon being inspected or as a routing bridge that routes the data received from the other nodes towards the destination. The nodes will need to update their routing information frequently which leads to a peer-to-peer broadcast where any node can communicate with its neighboring nodes when required. The peer-to-peer broadcast scheme causes very high traffic in the network, and it is usually implemented via flooding store-and-forward mechanism where each node forwards the data it receives from neighboring nodes on its way to the destination. The flood broadcasting usually suffers from data redundancy throughout the network, where a single node can receive the same data from many different peers. To increase the efficiency of data broadcasting and decrease the traffic in the network, an initial communication topology like trees, clusters [6] or star network where a central node is responsible of sending/receiving nodes from its direct neighbors can be used instead of all-to-all broadcasting. Network Coding (NC) can help increase the efficiency of the sensor network significantly by minimizing the number of transmissions required to forward the packets from one node to the others. So instead of blindly storing and forwarding individual packets, NC merges the packets together and forwards the combined packet to the neighboring nodes.

The following simulation shows the impact of NC versus the traditional store-and-forward mechanism in a mesh-based WSN. Figure 1 shows a network consisting of 15 nodes and a destination where the sensors need to deliver 4072KB of data. For the rest of the simulation, the nodes will represented using a small circle while the destination is shown as a small square. The actual connection speeds between the nodes are generated randomly with a maximum physical layer bit rate (PHY) of 1000kbps. The transmission channels are considered to be ideal and are represented using various line width, where wider lines mean higher speed.

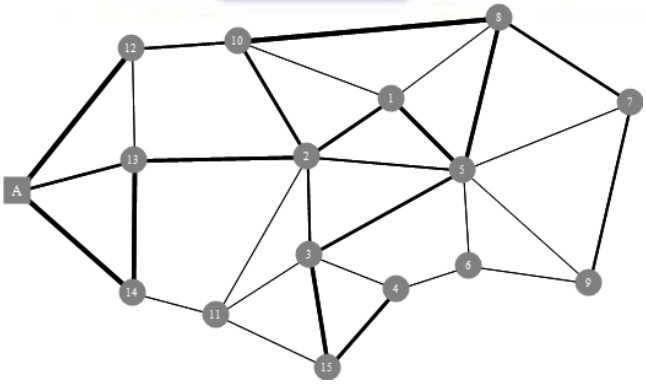

Figure 1. Mesh-based Sensor Network (n=13)

The generated links in the network shown in Figure 1 vary from 77Kbps to 993Kbps. Table 1 shows the result of the simulation without applying the network coding and after applying it on the network.

Table 1. SIMULATION WITH NETWORK CODING COMPARISON

	Without Network Coding	With Network Coding
Amount of aggregated data (KB)	31600	28328
Time until first delivery (Seconds)	28	23
Time required to deliver the data (Seconds)	338	318

The aggregated data represents the total amount of data exchanged between all the nodes until it reaches the destination. By implementing NC, the amount of aggregated data decreased 10.35% down from 31600KB to 28320KB. This decrease in the amount of data exchanged means lesser number of transmissions between the nodes which increases the energy efficiency of the network. Using NC also contributed to faster delivery of the data where it reached the destination 5 seconds earlier than without using it. It took the sensors 338 seconds to deliver all the data to the destination without implementing NC whereas; it required 318 seconds only to reach the destination after implementing it which means that NC can potentially help in increasing the efficiency of the network, therefore it will be used during the simulation for the rest of the paper.

III. PROBLEM STATEMENT

The nodes that have been attacked or compromised can no longer be trusted and their further presence in the network will lead to performance degradation. It is trivial for the network to detect a node that has been rendered completely dysfunctional by an attacker; such a node will not send or forward data any longer and thus, it can be easily avoided. The problem emerges mainly when the attacked node starts to misbehave by altering percentage of the data before forwarding it out to the surrounding nodes. In such case, it is not as easy for the surrounding nodes to detect the malicious one sending the corrupted data. While a node can check for the data it receives, it can never be sure that the sender is the attacker, because the sender might be just blindly forwarding the data. If a node is certain about the existence of an attacker, it can broadcast the attacker's address to the other nodes in the network to avoid further communication with the attacker, but the problem is that the other nodes can never be sure that the node sending the warning to them is not an attacker trying to dismiss clean nodes. This means that the decision to dismiss a malicious node from the network cannot be based on a single source. Figure 2 illustrates this scenario where the origin of the malicious data cannot be guaranteed.

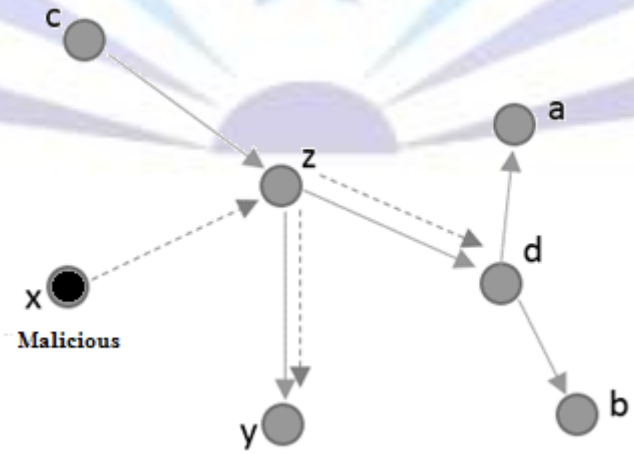

Figure 2. Malicious Source illustration

Node “x” is malicious and will start to send corrupted data to node “z” which in turn will start forwarding it to the other nodes in the network. Assuming that node “y” failed to decode the received data, this means that node “z” might be malicious or might be a clean node forwarding malicious data. To solve this problem a local decentralized voting technique is proposed. The technique is able to detect and dismiss the malicious nodes out of the network based on the decision of multiple neighboring peers. The technique uses a temporary candid node from the network to act as a local decision center to avoid fixing the location of the decision center as it might become the target of external attacks, and in case it is compromised, the technique will fail to work properly, also by making the decision center floating, it will be situated as close as possible to the area of attack and thus decrease the size of data overhead that needs to be exchanged between the nodes to detect and get rid of the attacker.

IV. PROPOSED TECHNIQUE

Every node in the network will build a short-timed trust value based on the behavior of neighboring nodes. If a node received corrupted data, it will decrease the trust value of the sender without taking further actions. After reaching the minimum threshold for the trust value, the node will start acting as a coordinator and will request the surrounds of the attacker to send their respective trust values. After receiving the trust values from the neighboring nodes, the coordinator will evaluate the probability of the suspect node of being compromised according to a preset function. If the node has been declared as an attacker, the coordinator will send a *<desync>* packets to the surrounding nodes to indicate the need to abandon it, and update their routing tables to redirect all the traffic to the center node if applicable. After receiving acknowledgment from the direct peers of the malicious nodes, the center node will finally abandon the malicious node as well. If the voting decision was not to abandon the suspect node, it means that the peers with the lowest trust values are suffering from transmission errors rather than active attacks. In this case, the center node will send an *<eval>* packet to the direct peers of the suspect node to indicate that the node is not malicious and that they should start reevaluating its behavior and it will also send *<sepa>* packets to the direct peers with the lowest trust value to instruct them that the node is not malicious but they should individually abandon it from their routing tables due to connectivity problems. The center node will take the decision based on a preset function with both the trust-level values received from the direct peers of the suspect node and the number of voting nodes as parameters. This way, a single node with an extremely low trust level will not be able to dismiss a suspect node from the network on itself. The technique is illustrated in Figure 3.

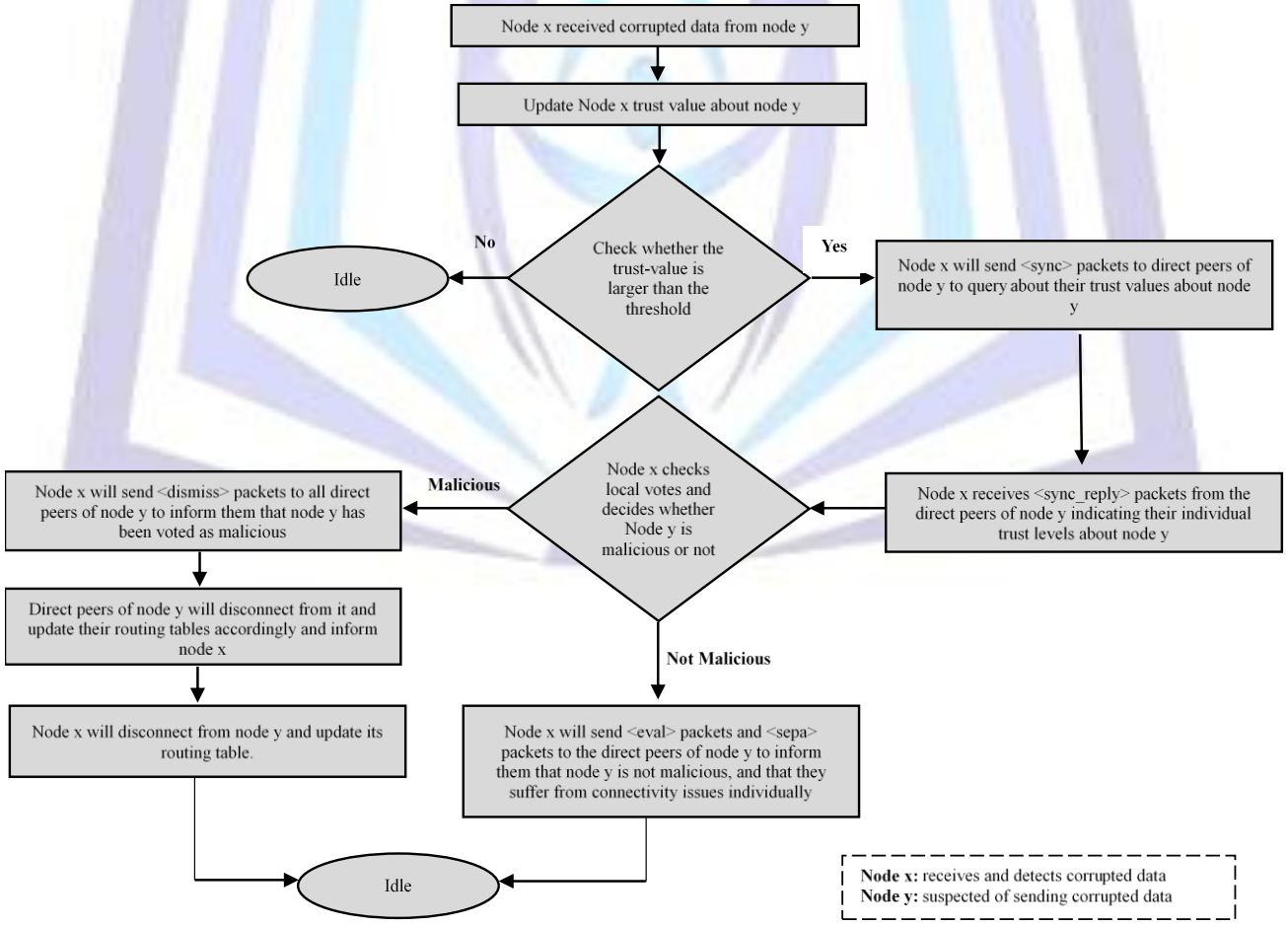

Figure 3. Proposed Technique illustration

To make the communication between the surrounding nodes of the suspect and the coordinator node protected, a secret sharing technique that is based only on the exchanges between neighbors nodes can be used [7]. The task of updating the routing tables for the nodes affected by the malicious node might not be simple and will depend on the topology of the

sensor network. Sometimes after the disposal of the malicious node. The coordinator may become out of the direct one hop range of affected nodes and due to the fact that the traffic cannot be redirect to it, the network will be split into two or more separated partitions. Therefore, after implementing the proposed algorithm, the network should be scanned for cuts and partitioning which can be accomplished using different algorithms such as the one suggested by Shrivastava et al. [8] and then an algorithm for reconnecting such partitions can be implemented [9].

V. SIMULATION AND RESULTS

The proposed technique is simulated using libCDN [10] with various sensor network layouts and network coding. The attack factor of malicious node is an indicator of the probability that the malicious node will corrupt the data, and it is generated randomly amongst the malicious nodes and is later fixed for the whole simulation scenario. The simulation results show the difference in the amount of data aggregated through the network and the time it took the data to arrive to its destination both before applying the proposed technique and after applying it. Figure 4 shows a scenario where two nodes are malicious with an attack factors of 40% and 70%. The links speed between the nodes varies from 8Kbps to 120Kbps.

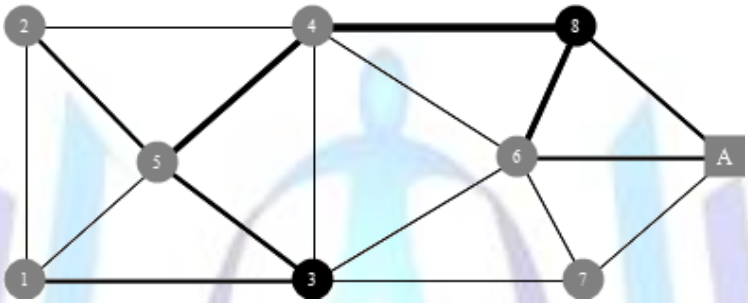

Figure 4. Scenario 1

The black-filled nodes (8, 3) are malicious nodes and the square node (A) represents the destination. The network should deliver 850KB of sensor data across to the destination. In this scenario, the malicious node (8) has the highest link speed of 120Kbps amongst all the network nodes and the link speed between node (8) and the destination (A) is 56Kbps which is very close to that between node (6) and the destination which will make up for the lost bandwidth.

Figure 5 shows the results of the simulation without applying the proposed technique and Figure 6 shows the results after applying the proposed technique.

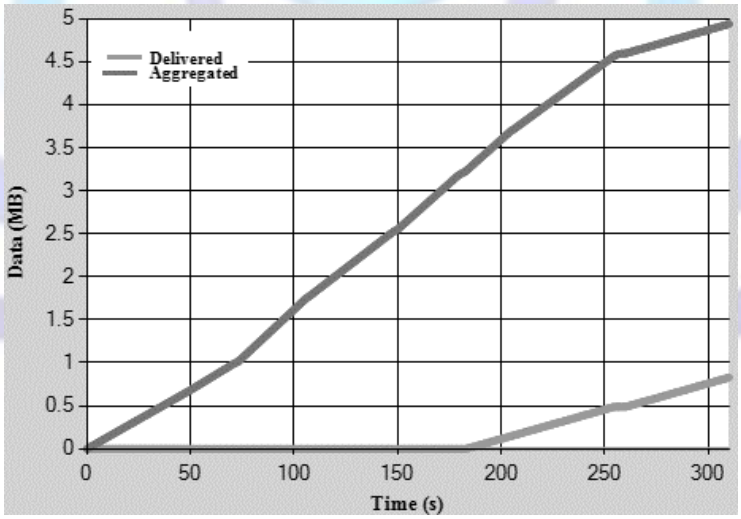

Figure 5. Scenario 1a (without proposed technique)

The amount of data aggregated through the network is 5056KB and it required the network 5.1667 minutes to finish delivering all the contents to the destination and it took 254 seconds for the first usable sensor data to start reaching the destination while after applying the proposed technique, the network took only 4.067 minutes to finish delivering the data to the destination, the amount of data aggregated was 4194KB and it took 145 seconds for the first usable sensor data to reach the destination.

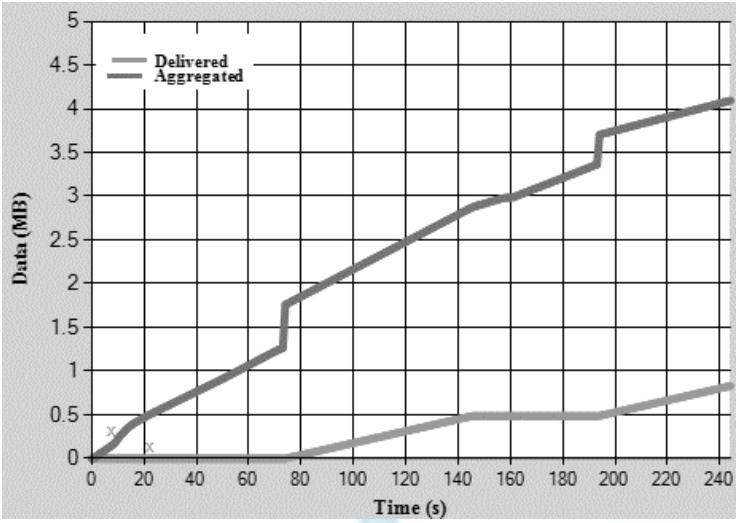

Figure 6. Scenario 1b (with proposed technique)

By inspecting both results, the removal of the malicious node helped decreasing the overall amount of data sent and forwarded by the sensors by 20.55% which will yield a more energy efficient network, also the first useable sensor data reached the destination 109 seconds earlier. The malicious nodes were detected and removed after 22 seconds from the starting of the simulation and are shown as a small crosses on the graph. By further inspecting scenario 1b, there are sudden increase in the amount of aggregated data that can be seen between the time 60 and 80 seconds and between 180 and 200 seconds. These sudden increase are the results of abandoning the malicious nodes and updating the routing tables, which in this case yielded a bandwidth increase due to reconnection with nodes having faster link speeds. Judging on the performance of network after applying the technique and removing the malicious nodes, it took the network another 226 seconds to deliver the remaining data, with an aggregation amount of 3540KB which yields an effective throughput of 15.663KB/s, but in scenario 1a, the effective throughput was 15.84KB/s which is slightly slower than scenario 1b.

The proposed technique helped increasing the throughput and the energy efficiency of the network by dismissing the malicious node(s) thus decreasing the amount of aggregated data through the network, but in other scenarios, the removal of malicious nodes or others with high error rate may increase the time required to deliver all the sensors data. Figure 7 shows another scenario where the removal of malicious/errands nodes may decrease the overall speed of the network for smaller amount of sensors data size.

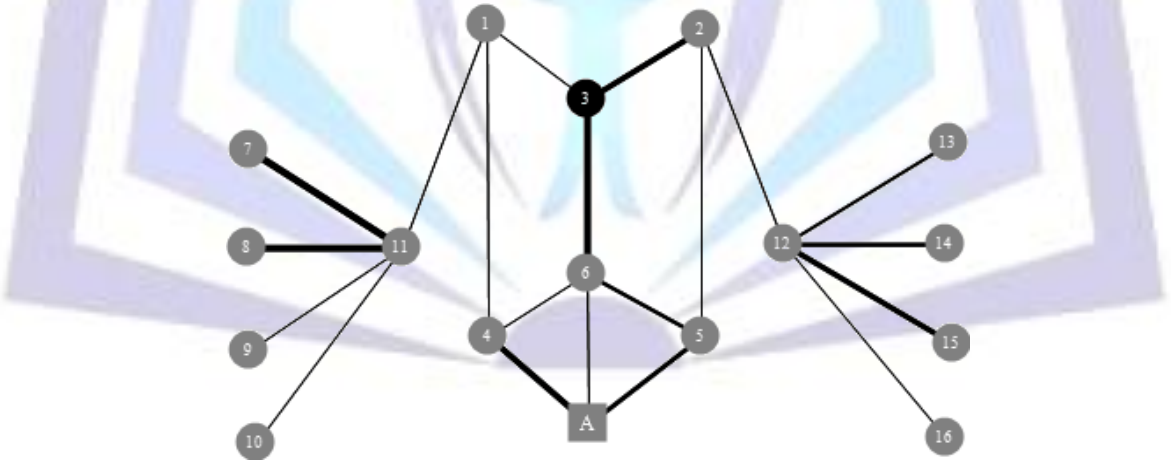

Figure 7. Scenario 2 (increased data size)

In this scenario, the link speeds vary from 12Kbps to 967Kbps and node 3 acts as an attacker with an attack factor of 30%. Both the left most nodes (7, 8, 9, and 10) and the right most nodes (13, 14, 15 and 16) will start sending each 750KB of sensor data to the destination (A), and then this amount of data is increased to study the behavior of increasing data on the proposed technique. Figure 8 shows that for smaller amount of sensor data, dismissing the malicious nodes is useless and might even cause a decrease on the overall network performance. Malicious nodes with locally high link speeds, bandwidth and low attack factors, although they corrupt data; it will still be better to temporarily keep them in the network. As the amount of sensors data increase, the necessity to remove such malicious node increases to maintain a high network throughput and performance.

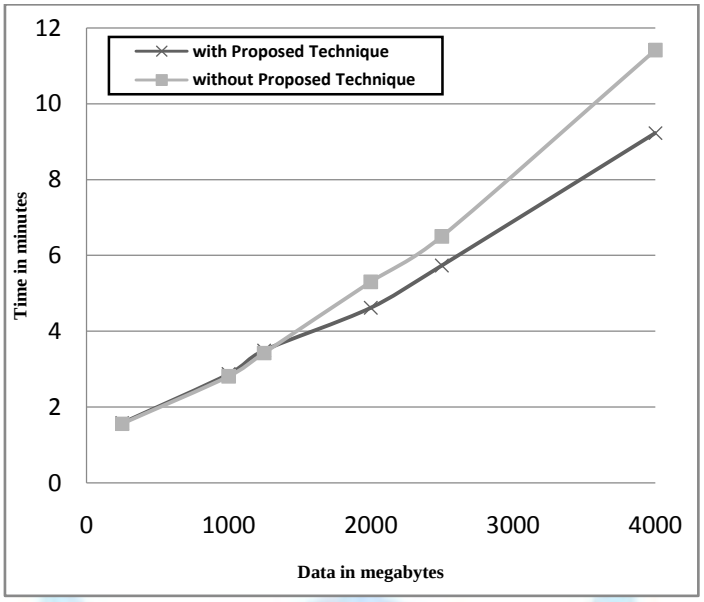

Figure 8. Simulation Results for Data Delivery Time

In case the proposed technique is used to dismiss the malicious nodes; as the amount of sensors data increases, the time required to finish delivering it decreases, but if the malicious node is kept in the network, the time required to deliver the data increases with the increase of the sensors data. Also, for amounts of data less than 1250KB, it will not be feasible to apply the proposed technique to get rid of the malicious node and that the network should continue delivering the data until the malicious node is later fixed and restored.

However, while the delivery time is negatively affected by applying the technique on smaller amount of data, the amount of aggregated data will still be decreased by applying it, thus increasing the energy efficiency of the network. Table 2 shows the time required to deliver all the data in both cases.

Table 2. TIME REQUIRED TO DELIVER DATA IN SCENARIO 2

Amount of Sensors Data (KB)	Time Required To Deliver All Data (seconds)		
	With Proposed Technique	Without Proposed Technique	Ratio (%)
1000	172	169	+1.74%
250	210	206	+1.904%
2000	277	318	-12.89%
2500	344	390	-11.79%
4000	554	685	-19.52%

The amount of time required to finish delivering the data with sizes varying from 1000KB to 1250KB ranges from 1% to ~2% before and after applying the proposed technique, which means that the dismissing of the malicious node is not effective for small data size and can increase the amount of time required to deliver all the data to the destination. Afterwards, the time required to deliver the data decreased with the increase of the data size.

Table 3 shows that the amount of aggregated data declines in respect to the increase of sensors data size. The links with high bandwidth that carry larger data are mostly affect by the increasing attack factors that in turn, destroys larger amount of data. Thus, as the amount of sensors data increases, higher percentage of decrease in the total aggregated data is achieved using the proposed technique.

Table 3. AMOUNT OF AGGREGATED DATA IN SCENARIO 2

Amount of Sensors Data (KB)	Amount of Data Aggregated (KB)		
	With Proposed Technique	Without Proposed Technique	Ratio (%)
1000	18920	18924	-0.02%
1250	21950	22066	-0.52%
2000	24854	31633	-21.43%
2500	29618	38068	-22.19%
4000	44201	57153	-22.66%

To study the effect of different attack factors on the given network, 7500KB of data were set to be delivered to the destination with an increasing attack rate that ranges from 10% to 90%. Figure 9 shows the simulation while using the proposed technique and without it.

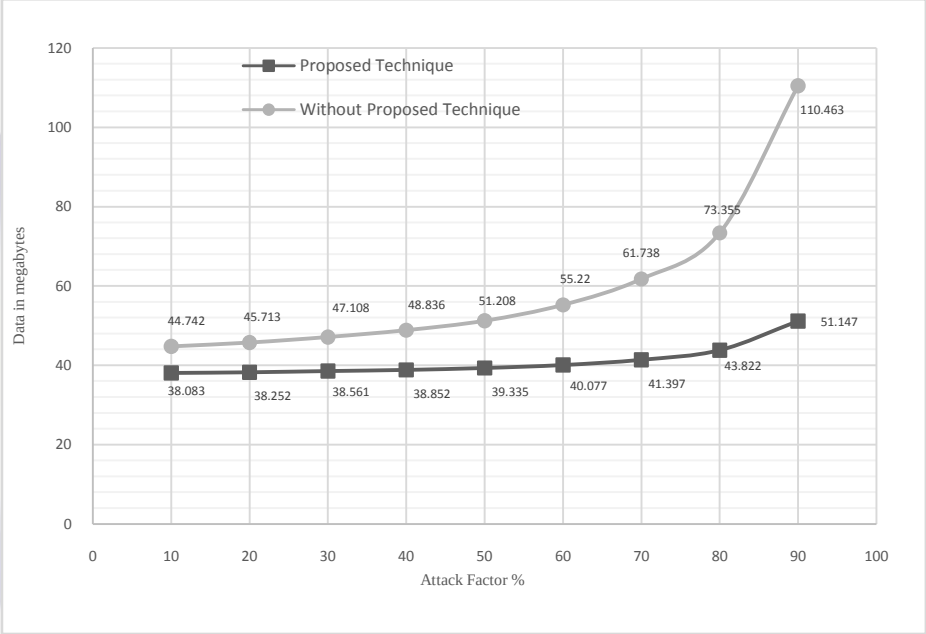

Figure 9. Simulation with increasing attack factor vs aggregated data

VI. CONCLUSION

This paper introduced a technique to detect and remove malicious nodes from the sensor networks using a local voting approach in an effort to differentiate between transmission errors and malicious nodes that have been compromised. The simulation results for the proposed technique showed that the effective throughput and the energy efficiency of the network can be increased with the dismissing of the malicious and errand nodes. However, it also showed that in a certain scenario with malicious node(s) processing fast link speeds and the network delivering small amount of sensors data between larger spans of time, it is feasible to only detect the presence of the malicious the node for later restoration without actually abandoning it from the network, especially if the amount of data corrupted by the node is not very high (e.g. node with low attack factor and link speed).

REFERENCES

[1] A. Gkikopouli, G. Nikolakopoulos and S. Manesis, “A survey on Underwater Wireless Sensor Networks and applications,” in *Conference on Control & Automation (MED), 2012 20th Mediterranean*, Barcelona, 2012.

[2] P. P. Mercier, D. C. Dal, M. Bhardwaj and D. D. Wentzloff, “Ultra-Low-Power UWB for Sensor Network,” in *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, Seattle, WA, 2008.

[3] S. Mao, M. H. Cheung and V. Wong, “An optimal energy allocation algorithm for energy harvesting wireless sensor networks,” in *IEEE International Conference on Communications (ICC)*, Ottawa, ON, 2012.

[4] M. Y. Malik, “An Outline of Security in Wireless Sensor Networks: Threats, Countermeasures and Implementations,” *Wireless Sensor Networks and Energy Efficiency: Protocols, Routing and Management,* 2013.

[5] R. Rajagopalan and P. Varshney, “Data-aggregation techniques in sensor networks: A survey,” *Communications Surveys & Tutorials, IEEE,* vol. 8, no. 4, pp. 48 - 63, 2007.

[6] D. Platz, D. Woldegebreal and H. Karl, “Random Network Coding in Wireless Sensor Networks: Energy Efficiency via Cross-Layer Approach,” in *International Symposium on Spread Spectrum Techniques and Applications, 2008. ISSSTA '08. IEEE 10th*, Bologna, Italy, 2008.

[7] M. Bertier, A. Mostefaoui and G. Tredan, “Low-Cost Secret-Sharing in Sensor Networks,” in *High-Assurance Systems Engineering (HASE), 2010 IEEE 12th International Symposium on*, San Jose, CA, 2010.

[8] N. Shrivastava, S. Suri and C. D. Toth, “Detecting Cuts in Sensor Networks,” in *Proceedings of The Fourth International Conference on Information Processing in Sensor Networks (IPSN)*, 2005.

[9] G. Dini, M. Pelagatti and I. Maria Savino, “An Algorithm for Reconnecting Wireless Sensor Network Partitions,” in *Wireless Sensor Networks*, Springer Berlin Heidelberg, 2008, pp. 253-267.

[10] R. A. Sadek, N. E. S. Eissa and M. M. Chahine, “Dynamic Routing Simulation in Content Delivery Networks,” *International Journal of Emerging Science,* vol. 2, no. 4, pp. 552-569, 2012.