Journal of Sensor Technology, 2014, 4, 48-58
Published Online June 2014 in SciRes. http://www.scirp.org/journal/jst
http://dx.doi.org/10.4236/jst.2014.42006

Wormhole Attack Behaviour in Monte-Carlo Localization for Mobile Sensor Networks

Vennam Ratna Kumari[1], Aitha Nagaraju[2], Gaurav Pareek[2]

[1]Nigama Engineering College, Karimnagar, India
[2]Central University of Rajasthan, Ajmer, India
Email: ratna@nigamacollege.com, nagaraju@curaj.ac.in

Received 16 February 2014; revised 17 March 2014; accepted 24 March 2014

Abstract

Localization is the basic requirement for network management in Wireless Sensor Networks as it helps nodes find their absolute position coordinates and in gathering information relevant to their locations. A localization algorithm has to be dynamic, scalable and should not impose high computation or communication overhead. The localization systems are also prone to attacks. We target a localization scheme for mobile sensor networks called Monte-Carlo Localization, which study its behavior under the most dangerous attack on localization called Wormhole Attack, also known as Collusion Attack and propose a modified algorithm that can help the localization system retain its accuracy level even in the presence of attacks. Our algorithm has communication cost almost equal to that of original localization algorithm (in this case MCL) in the absence of attacks.

Keywords

Localization, Sensor Networks, Collusion Attack

1. Introduction

Localization is a process by which the nodes in a sensor network find their location coordinates Global Positioning System (GPS) attached to an object determines the object's location. But nodes in a sensor network may be deployed in a GPS denied environment or the cost constraints may not allow every sensor node to possess a GPS device. In such cases, the nodes not knowing their location coordinates calculate their locations with the help of a few GPS enabled nodes (also called Anchor Nodes) or nodes that are strategically deployed at specific and known locations. Localization also helps nodes gather information relevant to their locations. A localization scheme has to be low cost, dynamic, feasible and considerably accurate for finding out the locations of the sensor nodes. Localization can be either range-based or range-free depending upon whether the algorithm uses the approximate range between the normal nodes and GPS enabled nodes or not respectively. The Monte-Carlo Localization (also called SMC based localization or MCL) based on Sequential Monte-Carlo Method [1] is widely studied for localizing mobile sensor nodes in a sensor network. The localization systems are prone to attacks and so is SMC-based localization. The attacks affect the expected accuracy level of the Localization Algorithms and the sensor nodes therefore believe that they are at some other position than they actually are or are supposed to be. So the aim of this paper is to study the behavior of MCL under attacks and propose a modified algorithm for securely localizing nodes in a sensor network using SMC method. Our algorithm has communication overhead equal or comparable to that in original MCL in the absence of attacks.

1.1. Introduction to MCL

In MCL, the localization problem is modeled as a non-linear stochastic process and the problem is to converge the probability distribution of the node being at a particular position given the previous positions $\left(l_{t-1}^1, l_{t-1}^2, l_{t-1}^3, \cdots, l_{t-1}^n\right)$ of the node itself and the beacon signals observed in this step $\left(o_1, o_2, o_3, \cdots, o_n\right)$. The algorithm consists of three steps namely Initialization, followed by iterative Prediction and Filtering. In initialization step, nothing is known about the position of the nodes (except their maximum velocities) and hence the distribution of nodes is supposed to be completely random in the whole region. Then in the next step prediction, the samples of the nodes positions are calculated depending upon the previous samples, *i.e.*, we calculate $R_{\text{filtered}} = l_t^i \mid l_t^i$ where $l_t^i \in R$ and $p\left(o_t \mid l_t^i\right) > 0$. Here N is the number of samples (size of the sample set) required for drawing the final position estimate. Now, filtering is done as the last step in which the invalid samples are eliminated from the set of samples depending upon whether the samples receive any one hop or two hop beacon signals or not. That is, now we calculate the complete set of samples by considering the filtered sample set until we get enough samples for that position (set to 50 out of total 80 by Hu and Evans). The weighted average of the samples is calculated and the result is the final position estimate of each node. If the algorithm is not able to calculate enough samples in one go, the same steps are repeated for the relaxed speed condition. If the algorithm still fails to calculate the samples, the algorithm does not try further to get the valid samples.

1.2. Introduction to Wormhole Attacks

This type of attack involves relay of the beacon messages from one place and replay of the same at some other part of the network giving the sensor nodes in the latter region an illusion that they are in the former region. This type of attack arises when the attacker cannot compromise the shared secret key. In this, the two wormhole endpoints are deployed in the two distant parts of the network. One to capture the beacon signals broadcasted in one region and the other to replay the relayed messages in its region. All the nodes that fall in the communication range of the end-point receive the signal and localize themselves incorrectly. Obviously, the number of neighbors of a node increase due to attacks some papers like in [2] take this as a basis for detecting the presence of attacks using connectivity information.

We shall discuss detailed analysis of the attacks in the later sections. In the presence of wormhole attacks, the scenario in **Figure 1** and **Figure 2** would apparently be like scenario in **Figure 3**. So, the one-hop neighbor at the other end of the wormhole link will always have same orientation with respect to the node under attack irrespective of the actual orientation.

2. Related Work

Several improvements were done on MCL in both sampling efficiency and accuracy. Some attempted to improve the accuracy of MCL even for static networks making the algorithm and more generally, the Monte-Carlo technique, more versatile. Monte-Carlo Boxed (MCB) proposed in [3] improves the sampling efficiency of the MCL algorithm by constraining the area to sample from, to a sample and a more accurate sample box. The Anchor Box is a region where the beacon ranges of beacon nodes overlaps and the sample box is the square of side 2 $v_{\max}$. The sample box comes into picture only if the location estimates are highly consistent. The sample box resets to the anchor box even upon slight location inconsistencies. The anchor box resets to the whole deployment area when such location inconsistencies arise with respect to beacons. Attempting to increase the accuracy level of the Monte-Carlo localization algorithm for mobile sensor networks, [4] includes two algorithms that outclass each other in different scenarios.

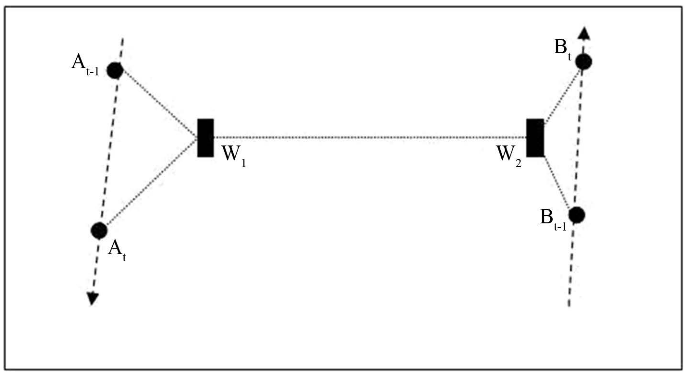


Figure 1. Possible scenario 1.

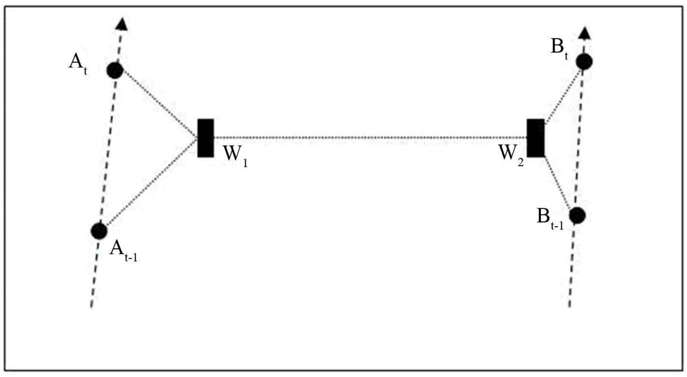


Figure 2. Possible scenario 2.

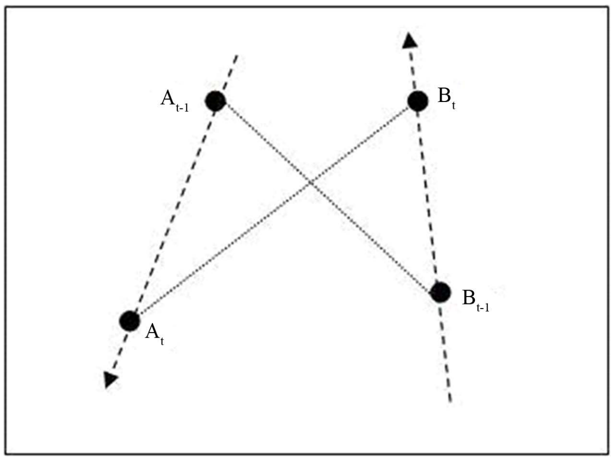


Figure 3. Apparent scenario.

The researchers in [4] proposed many improvements over the MCL simulator. First, both MSL and MSL* modify the sampling procedure enabling the nodes to work for even the static networks. Any node draws samples from the three classes of nodes namely one hop beacons, two hop beacons and one hop neighboring normal nodes. Second, the node uses the location estimates of those nodes (one hop normal nodes) whose location estimates are more consistent than it. The node does this by calculating its closeness value and comparing it with the closeness values of neighbors. The closeness values are calculated by calculating the weighted deviation of all the samples of the nodes location from the current location of the node. This gives the quality of location estimates of a node which is broadcasted by each normal node along with the packets containing the location of it. Third, they modified the sampling procedure such that the weights had values greater than a threshold which meant fast convergence of probability. However, they clearly specify the limitations of both the schemes. MSL* can work accurately even when the node speeds are very low but it involves high communication over-head. MSL is an optimal solution to the mobile sensor localization with high accuracy. The algorithm proposed in [5] works for both sampling efficiency and accuracy simultaneously. SecMCL [6] uses asymmetric key signature scheme for authenticating the nodes in the network. The scheme however does not address the Wormhole Attacks where even these signed messages (direct and indirect beacon signals) can be relayed from one part of the network to another and then replayed to create an unauthorized effect and can further affect the localization accuracy. The paper therefore proposes a scheme for securely discovering the neighbors that does not detect Wormhole Attacks. All the above approaches have been implemented, tested and analyzed by modifying the MCL simulator provided by Hu and Evans. Now we briefly review the existing Secure Localization algorithms and some Secure Neighbor Discovery approaches. There is a rich literature on secure localization in wireless sensor networks [7] [8]. All these approaches do not match up with the dynamic and unpredictable nature of Random Waypoint Mobility Model as they impose all or some of these requirements on the nodes in the networks fast processor for doing XOR operation efficiently, exact location estimates before attacks, tight clock synchronization, directional antennas [9]. All these assumptions do not hold with the system model for which MCL was developed. Use of directional antennas (as in [9]) can avoid wormhole attacks alone but their cost limits the scalability of the sensor networks. Secure Neighbor Discovery Algorithm in [2] makes the network nodes search for the forbidden substructures in the unit-disk graph (UDG) around a node. Finding a forbidden substructure flags the presence of a wormhole in the network. The verification of the results shows that the algorithm is 100 percent accurate with 0 percent false positives. But when the nodes move randomly, the algorithm has been shown to have deterioration in accuracy. Also, the UDG model is quite simple as compared with the requirements in RWP motion of nodes in the networks. The problem is common in almost all kinds of localization schemes in presence of adversaries. What would happen if a wormhole end-point claims to be a legitimate one hop neighbor? The error multiplication is even more dangerous when in MSL's modified sampling procedure because the node to be localized trusts comparatively more nodes (moreover normal nodes) to compute its location. So, the problem has to be solved a priori sampling. MCL is a localization method in which the attacker succeeds in introducing substantial error in location estimates only as long as the attacks persist, meaning that once the nodes to be localized escape the area under attacks, the accuracy level can be retained. But the attack has to be detected and avoided well before the stable step (set to 50 in [1]) to improve overall accuracy of the sensor network under attack.

3. Assumptions

3.1. Network Model

There are normal nodes and beacon nodes. Both the normal nodes and beacon nodes may be mobile. The nodes in the network do not know anything except their maximum velocity, current velocity, current time-stamp (up to hundreds of microsecond's precision) and their transmission ranges. The nodes need not have tightly synchronized clocks, the clocks are required to be loosely synchronized (*i.e.* having a time difference of not more than 1 sec). How this loose synchronization is achieved between the clocks is achieved is out of scope of the paper. Every node has a shared secret key for Symmetric Encryption. All these nodes are distributed randomly in a 500 $\times$ 500 cm^2 region and travel according to RWP mobility model in successive steps.

3.2. Attacker Model

We assume a scenario in which the two wormhole end-points (A and B) are placed in two distant regions in the network. We assume a wired connection between A and B. The end-point A collects all the signals (from both beacon nodes and normal nodes) and relays them through the wired connection to B and then B replays all of them in its region. We assume that the attacker is not able to compromise the key. Also, we confine ourselves to false neighborhood generation capability of the wormholes, *i.e.* we assume that the attacker does not wish to gain by selectively delaying or discarding the packets. The communication ranges of the wormhole end-points are finite. The reason behind selecting this model particularly is that the nodes following RWP model tend to move from the corners of the deployment area to its center [10]. So, when there is a sparse density of legitimate beacon signals, nodes are more likely to be localized incorrectly. The robustness of the secure localization algorithm we propose here will be tested even in the presence of small number of beacons. Also, our purpose of analyzing area under attacks and proposing a solution independent of the nodes outside the hostile area would be fulfilled with this.

4. Proposed Work

The algorithm presented here is of resilient type and enforces no or little extra communication overhead in the absence of attacks. For achieving this, the node has to initiate discovery of collusion only when an attack exists. As mentioned in [3], a node in a network can maintain both anchor box and sample box. We observe that upon encountering collusion, the node is no more capable of building bounding box less than the deployment area. Clearly upon, receiving the location information from the node at (x_4; y_4) and applying the following relations:

$$x_{\max} = \max_{j-1}^{n}\left(x_j - r\right), x_{\max} = \min_{j=1}^{n}\left(x_j + r\right) \quad (1)$$

$$y_{\max} = \max_{j-1}^{n}\left(y_j - r\right), y_{\max} = \min_{j=1}^{n}\left(y_j + r\right) \quad (2)$$

n and *r* being the number of anchor signals received and communication range of the nodes, we get from **Figure 4** $x_{\min} > x_{\max}$ and/or $y_{\min} > y_{\max}$. Now the node resets its anchor-box to the whole deployment area. This triggers a series of checks for neighborhood verification that we discuss in the upcoming sections. In the rest of the paper, we would refer to this preprocessing step as anchor-box-resetting test.

4.1. Overview of the Algorithm

```
Anchor Box(last_step) = Anchor_Box(this_step);
Synchronize Clocks;
Request Beacon nodes their Locations;
Collect Response;
while all the response messages are processed do
      Calculate Range;
        end
Build Anchor _Box(this step);
if Anchor _Box(this _step)==Deployment Area then
while all one-hop anchors are not processed do
    Apply Preliminary Checks;
end
Discard links that fail Preliminary Checks; Categorize beacons into 2 groups;
while i # 2 do
      Build Anchor_Box$_i$(this step);
      if Anchor Box$_i$(this step) # Deployment Area then
            Apply Final Checks;
        end
    end
end
MCLocalization();
Algorithm 1 : Overview of the Proposed Scheme
```

4.2. Communication Protocol

In **Figure 5**, we present the location request and response messages broadcast by normal and beacon nodes respectively. Along with broadcasting location request and response, the nodes also store their transmission times that help nodes synchronize their clocks later, if required. Now if the node carries out the Anchor-Box-Resetting test and the test does not hold true, the node will initiate the Preliminary Checks followed by Final Checks. For carrying out the preliminary checks, the node has to calculate the approximate range between the node and the corresponding set of neighboring beacon nodes. For calculating the range the nodes employ the timestamps stored in the previous step and the timestamps in the ranging step. Our ranging step is inspired by [11]. The use of the random nonce here ensures the freshness of the reply. The REQ and RES messages help the nodes in the

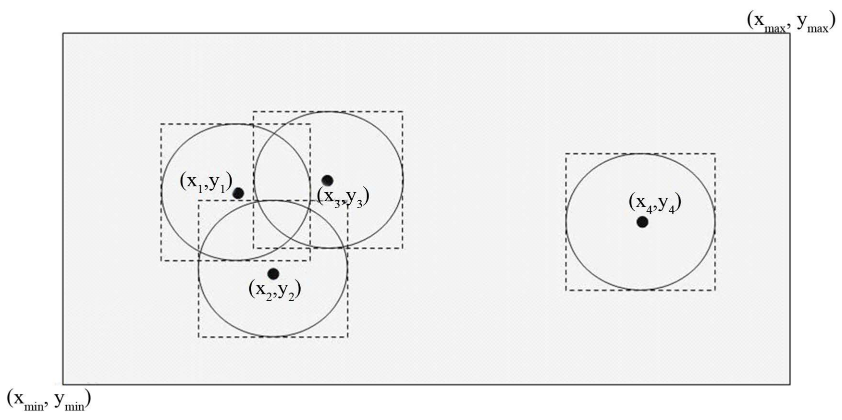


Figure 4. Resetting of anchor box (shaded region) due to colluding node (x_4; y_4).

N: Normal Node	$N \rightarrow$ Region: REQ ∥ ID_N
S: Seed Node	$N : t_{REQ}^{N} :=$ Sending time of REQ
	$S : t_{REQ}^{S} :=$ Reception time of REQ
	$S \rightarrow$ Region : REP ∥ ID_S ∥ loc_1
	$S : t_{REP}^{S} :=$ Sending time of REP
	$N : t_{REP}^{N} :=$ Reception time of REP

Figure 5. Normal node requests for beacon messages and beacon nodes respond back.

network loosely synchronize their clocks. Now, the range between the nodes can be calculated using the following relation:

$$\text{Range}_i = \left(\left(t_{\text{RNG}}^{S} - t_{\text{REQ}}^{S}\right) - \left(t_{\text{RNG}}^{N} - t_{\text{REQ}}^{N}\right)\right) \times s \tag{3}$$

For maintaining range between two normal nodes, the node will do:

$$\text{Range}_i = \left(\left(\left(t_{\text{RNG}}^{S} - t_{\text{REQ}}^{S}\right) - \left(t_{\text{RNG}}^{N} - t_{\text{REQ}}^{N}\right)\right) \times s\right) - \text{Range}(M, S) \tag{4}$$

where, s is the speed of the signal propagation. The radio waves used as transmission media between sensors travel with speed of light. So, the distance shortening attacks are not possible. Now the nodes receiving the beacon signals initiate a test which involves checking for links that have ranges exceeding the nodes communication range, *i.e.*, $R_i > N_{\text{RNG}} + \epsilon$ (where 2 is the ranging error + processing delay). Ranges as long as $2 \times N_{\text{RNG}}$ may be produced by the wormhole endpoints. A practical approach would be to include both processing and communication delay while measuring ranges between nodes. Now all the nodes M listening to the beacons that pass the above test broadcast the beacon message received by it as shown in **Figure 6**. Also the beacons that fail the above test are not considered by the node for estimating the nodes own position. Easily detectable colluding links are blacklisted by this preliminary check. Now, even if the presence of attacks is flagged by the anchor-box-resetting test, meaning that the colluding links are producing ranges less than or equal to the nodes communication range plus the constant 2, we initiate the final checks.

4.3. Final Checks

Since in the presence of attacks, a node has its anchor-box (comprising both one-hop and two-hop anchor nodes) equal to the deployment area, the box can be divided into several smaller anchor boxes (possibly equal in number to that of the total number of wormhole end-points *i.e.* two in our case). This categorization of anchor boxes is easy for one-hop anchors but difficult for two-hop anchor nodes. For a two-hop anchor node to be able to fall in a group of anchors, its distance from all the anchors in the box has to be less than or equal to 3 × Node_Range. The MCB is run for the two boxes so obtained. Now, the positions that are calculated, in the current step, if not equal to (250, 250) is broadcasted by the node using the following message:

M : Node that receives the beacon signals	M → Region : REP ‖ ID$_M$ ‖ IDs ‖ loc$_1$

Figure 6. Nodes broadcast the beacon signals received in current step.

$$N \to \text{Region} : \text{CHECK} \parallel E_K\left(N_{S1}, \text{es\_pos}_1, \text{es\_pos}_2\right)$$

where N_{S1} is another nonce for ensuring data freshness and es_pos$_1$ and es_pos$_2$ are the position estimated over two bounding boxes calculated previously. **Figure 7** illustrates the messages exchanged and the time stamps. If two such communications occur successively between a normal node-beacon node pair, the beacon nodes now consider the position that is closer to them and perform classical scaling (a variant of Multidimensional Scaling) giving the following input distance matrix to it:

$$\left(\begin{array}{cccc|}
0 & A_{t-1}A_t & R_{i-1} & B_{t-1}A_t \\
A_{t-1}A_t & 0 & B_tA_{t-1} & R_i \\
R_{i-1} & B_tA_{t-1} & 0 & B_{t-1}B_t \\
\hline
B_{t-1}A_t & R_i & B_{t-1}B_t & 0 \\
\hline
\end{array}\right)$$

Here, A_{t-1} and A_t are the apparent position of the nodes (as calculated by the MCB performed over the two boxes previously), B_{t-1} and B_t are the known positions of the beacon nodes performing MDS (see **Figure 3**). The classical scaling gives all the coordinates as output. Based on these coordinates, the distance between each point is calculated as shown in **Figure 8**. Now, as stated earlier, in any case, due to the presence of wormhole link, the scenario would look like **Figure 2**. In the input distance matrix which we use for MDS, the ranging information may violate the triangle inequality required for MDS to work.

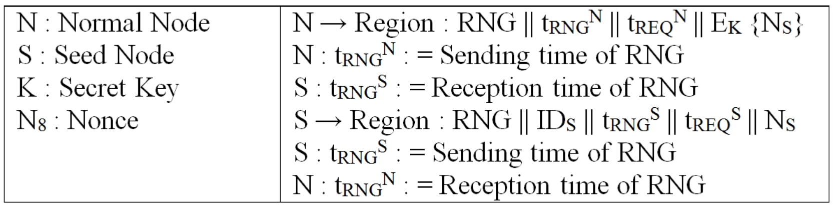
This is done by MDS by adjusting its stress factor. In case of an attack, the ranging information is longer than expected. So, the MDS would adjust the distance between successive positions of the nodes and provide a distance longer than the maximum velocity of the nodes (provided that the node estimates the apparent location with good closeness value). Since the distance travelled by a node cannot be greater than the maximum velocity of the node, the beacon node checks if both the distances (obtained by MDS and given as distance matrix entry) are less than the maximum velocity of the node. If the obtained distances and input distances both are less than the maximum velocity, or if the obtained distances and input distances both are greater than maximum velocity, the beacon node passes the test. The beacon nodes that fail the above test broad casts the message of the type:

$$S \to \text{Region} : N_{S1} \parallel E_K\left(ID_N \parallel \text{Blacklist}\right)$$

Upon receiving this message, the node IDN will blacklist the seed S, *i.e.* it will not consider the location estimate of beacon node S in further steps. After the node stops getting location response from the blacklisted beacon node, it removes the beacon node from the list of blacklisted nodes. Now, since we use the estimated positions for giving input to MDS, the rise in the number of false positives is obvious. In [4], a closeness parameter is defined as under:

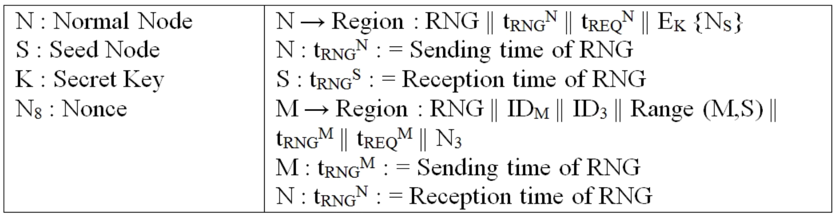
$$\text{Closeness}_p = \frac{\sum_{i=1}^{N} w_i \sqrt{\left(x_i - x\right)^2 + \left(y_i - y\right)^2}}{N}$$

The node's estimated position from the samples it draws is a measure of the quality of the location estimate of the node. Lesser the closeness value is, better the quality of location estimate is. Now the nodes that receive the blacklist signals from their respective one-hop beacon neighbors broadcast the same blacklist signals with their respective closeness values added in the beginning of the message. Now the nodes receiving the message compare their closeness values with the closeness value in the message. Nodes now consider only those messages that have closeness values less than the node's own closeness value, find the links that are blacklisted by the node but not by its neighbor with lower closeness value (false positives) and the links not blacklisted by the node but blacklisted by its neighbors with lower closeness value (false negatives). This way, the nodes with poor location estimates will also be able to secure the overall localization process and further improve the overall localization accuracy.



N : Normal Node	$N \rightarrow \text{Region} : \text{RNG} \Vert t_{RNG}^{N} \Vert t_{REQ}^{N} \Vert E_K \{N_S\}$
S : Seed Node	$N : t_{RNG}^{N} :=$ Sending time of RNG
K : Secret Key	$S : t_{RNG}^{S} :=$ Reception time of RNG
N_8 : Nonce	$S \rightarrow \text{Region} : \text{RNG} \Vert ID_S \Vert t_{RNG}^{S} \Vert t_{REQ}^{S} \Vert N_S$
	$S : t_{RNG}^{S} :=$ Sending time of RNG
	$N : t_{RNG}^{N} :=$ Reception time of RNG

Figure 7. Ranging operation.



N : Normal Node	$N \rightarrow \text{Region} : \text{RNG} \Vert t_{RNG}^{N} \Vert t_{REQ}^{N} \Vert E_K \{N_S\}$
S : Seed Node	$N : t_{RNG}^{N} :=$ Sending time of RNG
K : Secret Key	$S : t_{RNG}^{S} :=$ Reception time of RNG
N_8 : Nonce	$M \rightarrow \text{Region} : \text{RNG} \Vert ID_M \Vert ID_3 \Vert \text{Range}(M,S) \Vert t_{RNG}^{M} \Vert t_{REQ}^{M} \Vert N_3$
	$M : t_{RNG}^{M} :=$ Sending time of RNG
	$N : t_{RNG}^{N} :=$ Reception time of RNG

Figure 8. Ranging operation between two normal nodes.

5. Simulation Results

We modified the simulator by Hu and Evans and implemented Wormhole Attacks to analyze the performance of MCL. As stated earlier the following is the behavior of MCL under attacks in terms of increase in number of neighbors, fraction of nodes unable to fill their sample set with valid samples and rise in estimation error. We also modified the same simulator and implemented the proposed scheme and found that on an average, most of the colluding links were detected using preliminary checks. Rest of the attacks can be detected by using the final checks with high accuracy and less false negatives.

5.1. Results of Attacks

5.1.1. Increase in Number of Neighbors

Due to the attacker model assumed for experimental verification of the WRMCL algorithm, the number of neighbors that fall in the communication range of a node increases. The additional neighbors contain both one-hop anchor and normal nodes. The colluding links may also give rise to the increase in number of two-hop anchor neighbors. The graph in **Figure 9** shows the increase in number of neighbors of every node in the area under attack (on an average) in each step.

5.1.2. Rise in Fraction of Nodes Not Having Enough Valid Samples

Since the importance sampling strategy generates samples based on the node's previous location as well as the location of one-hop and two-hop beacons in the current step, the node will not be falsely localized until it has at least one legitimate anchor node in its one-hop two-hop neighbor list. However, the nodes will not be able to generate the samples that will localize the node at a particular position. So, in the absence of enough valid samples, the node will localize itself somewhere near the center of the region. And the algorithm will report: "Not Enough Valid Samples". **Figure 10** is the comparison of average fraction of nodes in the attacked area having such "inconsistencies" in position estimations in the presence of attacks and without attacks respectively.

5.1.3. Rise in Estimation Error

Estimation error is the distance between the estimated and actual positions of the nodes in the network. As stated earlier, the nodes encounter the condition of inconsistencies which contributes much to the average estimation error of each step. Also, the nodes that started estimating their positions in the attacked region, may get falsely localized at some other part of the region. **Figure 11** is the comparison of average estimation error in the attacked region in presence of attacks and without attacks.

5.2. Results on Applying the Proposed Algorithm (WRMCL)

Now, in this section we present the results of analysis of the WRMCL algorithm under colluding attacks and we

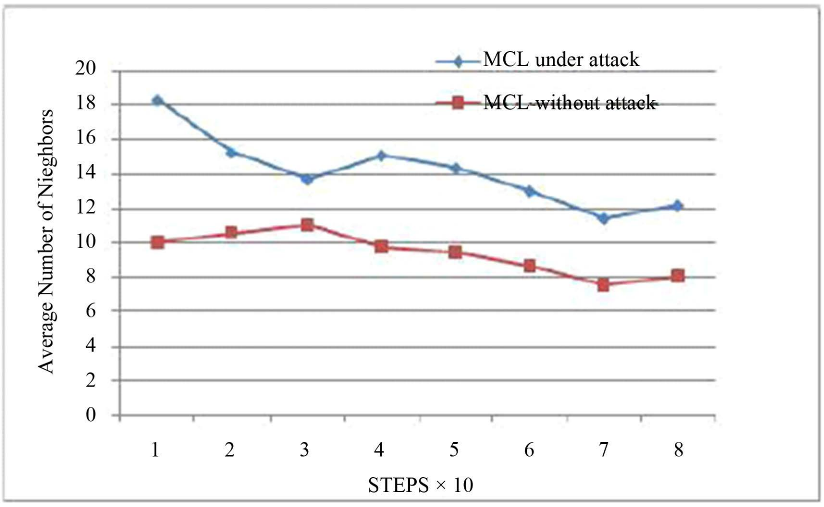


Figure 9. Increase in number of neighbors due to attacks.

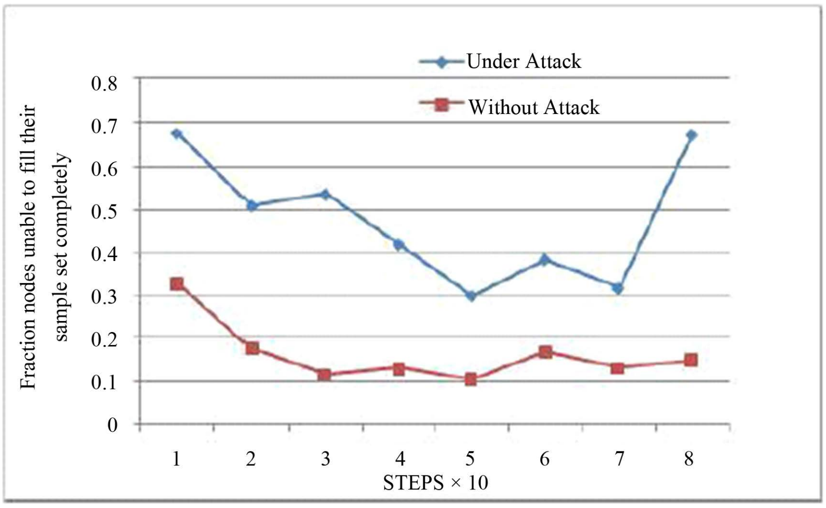


Figure 10. Rise in fraction of nodes having inconsistencies.

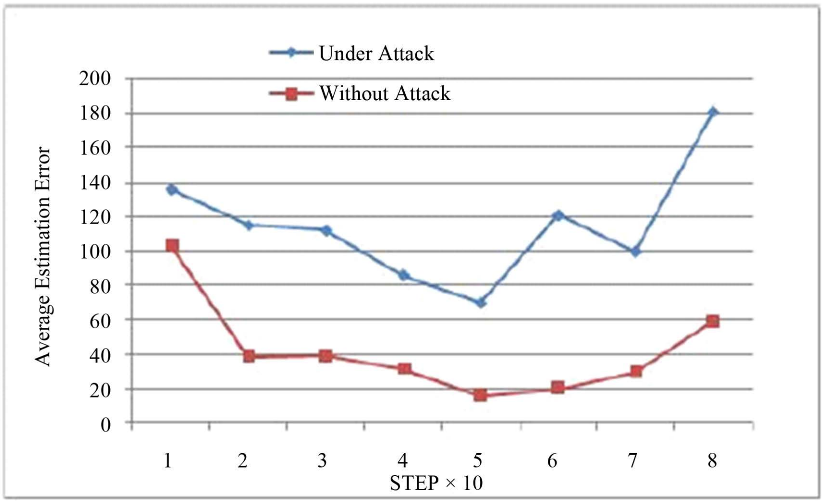


Figure 11. Rise in estimation error.

also show a comparative study about between MCL and WRMCL.

5.2.1. Average Estimation Error of the Area under Attack

The experimental results show that the number of steps it takes for the nodes, which started localization process in the hostile region, to converge to a minimum value decreases substantially. **Figure 12** is the comparative study of WRMCL with MCL in terms of the average estimation error produced in area under attack.

5.2.2. Overall Estimation Error

Figure 13 is the comparison between estimation error produced by the MCL and WRMCL in the whole region.

6. Conclusions and Scope

Wormhole attack is a dangerous external attack that can affect the accuracy level of any localization algorithm. Monte-Carlo Localization (MCL), a range-free localization scheme widely accepted for mobile sensors localization with both normal and beacon nodes moving, undergoes some serious degradation in its accuracy under these attacks. The approach we proposed detects and avoids getting location estimates from the non-neighbor

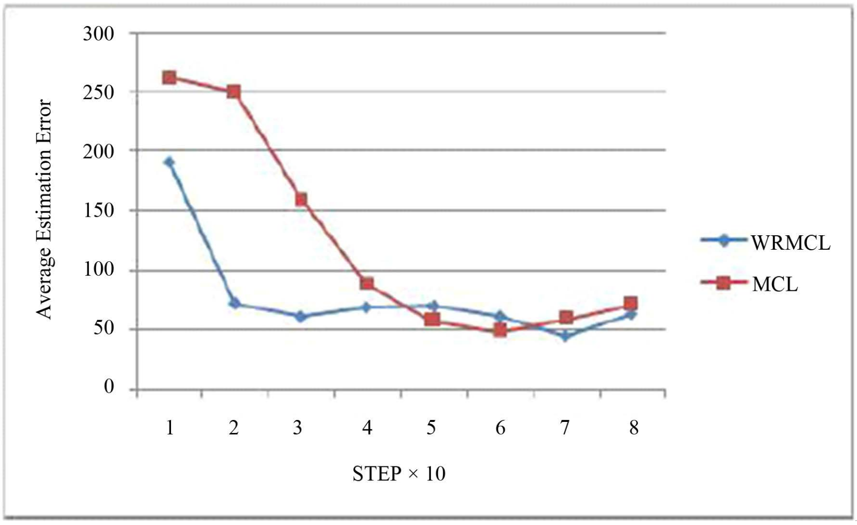


Figure 12. Improvement in estimation error in the area under attacks.

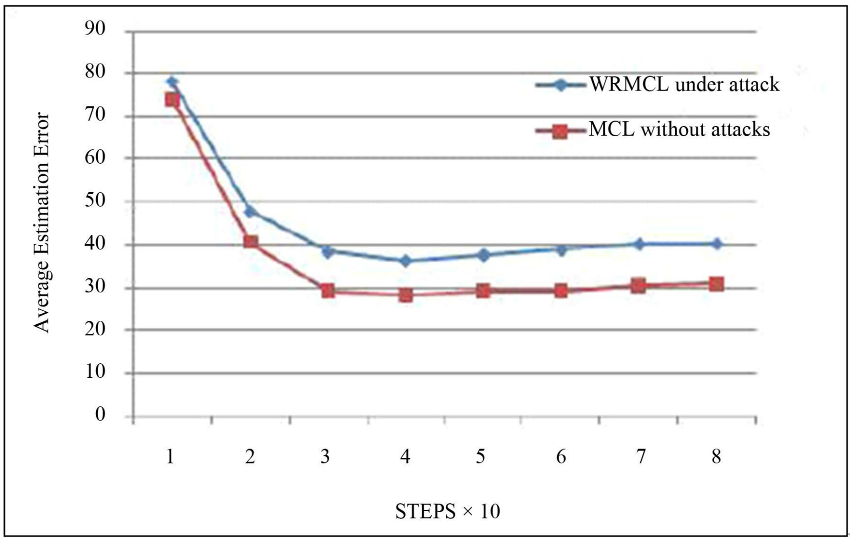


Figure 13. Comparison of overall estimation error of MCL and WRMCL.

beacon nodes using preliminary and final checks. The accompanying security analysis theoretically proves the appropriateness of the approach to MCL and demonstrates its security. The concrete analysis of false positives and false negatives of the algorithm, their variations with the factors such as stress factor, threshold, processing delay, exploring possibilities of using this scheme with other range-free localization schemes and a proof of concept implementation of the scheme remains as ongoing future work.

References

[1] Hu, L. and Evans, D. (2004) Localization for Mobile Sensor Networks. 10*th Annual International Conference on Mobile Computing and Networking* (*MobiCom* 2004), Philadelphia, 26 September-1 October 2004, 45-57.

[2] Maheshwari, R., Gao, J. and Das, S. (2007) Detecting Wormhole Attacks in Wire-Less Networks Using Connectivity Information. *INFOCOM*, 26*th IEEE International Conference on Computer Communications*.

[3] Baggio, A. and Langendoen, K. (2006) Monte-Carlo Localization for Mobile Wireless Sensor Networks. *Proc. Conf. Mobile Ad-Hoc and Sensor Net-works* (*MSN* 06), 317-328. http://dx.doi.org/10.1007/11943952_27

[4] Rudafshani, M. and Datta, S. (2007) Localization in Wireless Sensor Net-Works. *Proc.* 6*th International Conf. Information Processing in Sensor Networks* (*IPSN* 07), 51-60. http://dx.doi.org/10.1145/1236360.1236368

[5] Zhang, S.G., Cao, J.N., Chen, L.J. and Chen, D.X. (2010) Accurate and Energy-Efficient Range-Free Localization for Mobile Sensor Net-Works. *IEEE Transactions on Mobile Computing*, **9**, 897-910.

[6] Zeng, Y.P., Cao, J.N., Hong, J., Zhangt, S.G. and Xie, L. (2009) SecMCL: A Secure Monte Carlo Localization Algorithm for Mobile Sensor Networks. *Mobile Adhoc and Sensor Systems*, *MASS* '09.

[7] Capkun, S. and Hubaux, J.-P. (2006) Secure Positioning in Wireless Networks. *IEEE Journals on Selective Areas in Communication*, **24**.

[8] Lazos, L., Poovendran, R. and Capkun, S. (2005) ROPE: Robust Position Estimation in Wireless Sensor Networks. *The Proceedings of the* 4*th International Symposium on Information Processing in Sensor Networks*, *IPSN* 05.

[9] Hu, L. and Evans, D. (2004) Using Directional Antennas to Prevent Wormhole Attacks. NDSS04, San Diego.

[10] Bettstetter, C., Hartenstein, H. and Costa, X.P. (2003) Stochastic Properties of the Random Waypoint Mobility Model. *ACM/Kluwer Wireless Networks*, *Special Issue on Modeling Analysis of Mobile Networks*.

[11] Stoleru, R., Wu, H. and Chenji, H. (2011) Secure Neighbor Discovery in Mobile Ad Hoc Networks. 8*th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems*, 35-42.