

SLAT: Secure Localization with Attack Tolerance

Matthew Pirretti, Narayanan Vijaykrishnan, Patrick McDaniel, and Bharat Madan
The Pennsylvania State University

Abstract

Accurate and secure localization is essential to the correct operation of many applications of sensor networks. However, existing methods lack concrete security mechanisms or are not resilient to beacon node compromise. We address the limitations of present approaches in this paper through the Secure Localization with Attack Tolerance (SLAT) protocol. In SLAT, non-beacon nodes estimate their positions by calculating the intersection of multiple authenticated beacon messages. Message authentication prevents wholesale beacon location report forgeries. To combat compromised beacons, we develop and analyze a location reporting algorithm that ensures that compromised beacons have little ability to affect location estimates. Moreover, the degree to which a malicious location report affects an estimate is inversely proportional to its distance from the true location (as reported by the majority of properly operating beacons). We evaluate the protocol via simulation within a range of sensor networks and protocol parameters. Our results indicate that even large numbers of compromises only nominally affect location estimates. For example, we show that compromising 40 out of 200 nodes in a simulated environment only increases the average location estimate error from 3 meters to 5 meters. Such results apply across a wide range of networks, topologies, and hardware platforms.

I. INTRODUCTION

Localization is defined to be the process by which nodes in a network determine their physical locations. Localization is particularly important in sensor networks, where many applications require each sensor node to know its location. Consider the role localization would play in the following two sensor network applications. In a target tracking application, a sensor network monitors and tracks vehicles and personnel traversing through the network. The network's owner is given information characterizing any targets of interest. Such information includes the target's location, speed, bearing, etc. In order for the network to reliably generate such information for the user, each sensor node in the network must know its own location. Thus, localization is an absolute requirement for sensor network target tracking. Now consider a scenario where sensor nodes are actively monitoring industrial machinery for the purpose of providing early detection of part failures. If the individual nodes have been localized, then they are able to alert the machine's technician as to where the faulty part may be. Thus, localization greatly adds to the utility of the monitoring application.

There are two broad approaches to performing localization in a sensor network. In the first approach each node is explicitly given its location. In the second approach the sensor nodes autonomously determine their own location by utilizing a positioning system such as GPS. The huge time investment required of the first approach precludes its usefulness to all but trivially small networks. Thus we have only consider networks where GPS is used to perform localization.

Ideally, each node would be given its own GPS receiver, enabling it to independently determine its position with great accuracy. Unfortunately, the high costs of GPS technology are at odds with the desire to minimize the cost of individual nodes. Thus, it is only feasible to fit a small portion of all sensor nodes with GPS receivers. These GPS-enabled nodes, called *beacon nodes*, provide position information, in the form of *beacon messages*, for the benefit of *non-beacon nodes* (i.e. nodes without GPS capabilities). Non-beacon nodes can utilize the position information furnished from multiple nearby beacon nodes to estimate their own positions, thus amortizing the high costs of GPS technology across many nodes. An example of the GPS-based paradigm of localization is pictured in Figure 1. In this figure two beacon nodes, labeled B1 and B2, are broadcasting beacon information to five non-beacon nodes, labeled 1 through 5. B1's beacon information is being received by nodes 1, 4, and 5. B2's beacon information is being received by nodes 2, 3, and 5. Notice that node 5 can utilize the beacon information from both B1 and B2 to attain higher certainty in its position relative to nodes 1 through 4.

However, given the hostile environments in which sensor nodes are expected to be deployed, it is crucial to make sensor network algorithms robust against attacks. Unfortunately, most localization schemes have not been designed with security in mind and thus can be trivially abused by a malicious adversary. For instance, an adversary can

This research is sponsored by the Defense Advance Research Projects Agency (DARPA), and administered by the Army Research Office under Emergent Surveillance Plexus MURI Award No. DAAD19-01-1-0504. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies. This work is also supported by MARCO/DARPA GSRC Grant and NSF CAREER 0093085.

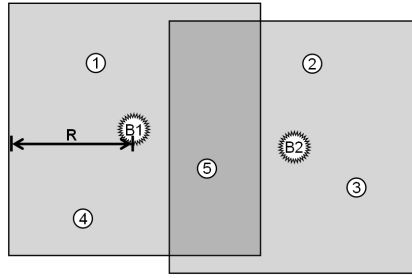


Fig. 1: Example of GPS-based localization.

perform what we call *location poisoning*, an attack designed to undermine localization. Localization schemes which do not authenticate beacon nodes enable an adversary to carry out location poisoning simply by forging erroneous own beacon messages. These malicious beacon messages can be designed to be indistinguishable from legitimate beacon messages. As a result of location poisoning, non-beacon nodes will experience a vast amount of error in their position estimates. Further, the adversary can readily launch a Sybil attack, by masquerading as multiple legitimate nodes, thereby compounding the effects of an attack [1]. The adversary can utilize a Sybil attack to generate as many false beacon messages as is necessary to counteract legitimate beacon messages. Consider how an adversary could perform location poisoning in our two example sensor network applications. Consider the target tracking application from a military context where a sensor network is being used to provide situational awareness of enemy troop positions. Enemy troops could use location poisoning in order to deceive the network into believing they were located at a vastly different location. In this context such an attack could readily cost lives. Now consider the monitoring example, where a disgruntled employee plays the role of adversary. A malicious user could use location poisoning to mislead the network into believing that the faulty part was located at a vastly different location, wasting the machine's technician time by causing him/her to look for a problem where there was none.

Clearly it is of paramount importance to design localization to be resilient to location poisoning. As such, we define *secure localization* as being a localization scheme which is resilient to location poisoning. We contend that there are two critical components to making localization secure: 1) authentication of beacon information and 2) resiliency to compromised beacon nodes. The authentication component is crucial to prevent non-beacon nodes from creating beacon information. However, authentication alone is not enough to secure localization. In sensor networks, legitimate nodes (including beacon nodes) can be compromised [2]–[7]. Having once been legitimate nodes, compromised beacon nodes will possess all of the proof of authentication necessary to generate beacon information. Therefore, it is critical for a secure localization scheme to authenticate beacon information and be resilient to compromised beacon nodes.

In this paper we present our approach to localization, which we call Secure Localization with Attack Tolerance (SLAT). SLAT is designed to both prevent the adversary from generating unauthorized beacon information and to be resilient to legitimate beacon nodes which have been compromised. SLAT exploits past work in sensor network key management to perform authentication. Specifically SLAT builds upon the LEAP key management protocol [8]. To be robust against compromised beacon nodes, SLAT builds upon fault tolerant sensor sampling as described in [9]. By utilizing these two separate components in a single system, we have been able to design a localization scheme which is very secure.

Our work on secure localization is novel in that it is one of the first to consider both authentication of beacon information and resiliency to compromised beacon nodes. Further, this is the first work to both consider the effects of location poisoning upon the entire network and a localized region of particular interest to the adversary. Our method is also novel in that it does not require the existence of additional information such as signal strength, time of packet arrival, etc. to enable non-beacon nodes to estimate their proximity to beacon nodes. Since our scheme does not depend upon these tools we can avoid communications issues such as noise, obstructions, etc. in environments where such techniques would be problematic. However, in less hostile environments our technique can readily make use of these tools to improve the certainty of non-beacon node position. Additionally, our method can readily adapt to the position error inherent to utilizing GPS. Other works have assumed beacon nodes are given perfect location information by their GPS receivers. However, GPS receivers do not provide pinpoint accuracy; most GPS receivers can only guarantee accuracy to the order of 10 m [10]–[12]. Our technique can readily model this uncertainty in a beacon node's position.

II. RELATED WORK

Due to the great importance localizing sensor nodes, many localization schemes have been proposed specifically to fit within the constraints of sensor node hardware [13]–[17]. However, an adversary can utilize location poisoning to trivially undermine each of these schemes.

Only a few works have attempted to address secure sensor network localization [18]–[22]. The methods in [18] and [22] rely on authentication to perform secure localization; however, both methods are vulnerable to beacon node compromise.

The method proposed in [19], [20] requires that each sensor node maintain a globally consistent lookup table which enables non-beacon nodes to lookup their physical location using the beacon messages they can hear. This table could become unwieldy for large networks. Further, in a dynamic setting, where beacon nodes may be added or removed at any point in time, it seems problematic to be constantly updating the location tables of all nodes in the network.

The two methods proposed in [21] could be used in limited situations to perform secure localization. Unlike other schemes, both authentication of beacon information and resilience to compromised beacon nodes are addressed. In these schemes, non-beacon nodes form their position estimates by observing the signal strength of beacon messages. This method relies on the assumption that a message accompanying a strong signal strength must come from a node that is in close proximity. In an ideal environment, utilizing the additional information provided by beacon signal strength allows non-beacon nodes to refine their position estimates. The authors note that for this scheme to work, the error involved in performing estimation based upon signal strength must follow a known error model that must remain fixed. However, given that sensor nodes are deployed in chaotic environments such as battlefields and disaster areas, it seems likely that environmental factors (malicious and otherwise) could readily interfere with perceived signal strength, making the fixed error model assumption unrealistic. Additionally, this technique utilizes location references from multiple hops away, raising questions of its scalability due to the rapid increase in communication resulting from beacon messages propagating for multiple hops.

To make SLAT resilient to compromised beacon nodes, we utilize the fault tolerant sensor sampling method in [9]. This method was designed for the orthogonal issue of tolerating faulty sensors. For instance consider an application where highly accurate weather conditions are desired. To mitigate the risk of faulty equipment N different weather monitoring sensors are measuring the same weather conditions. Each sensor takes numerous measurements of the current conditions and provides a report in the form of a single interval. It is assumed that at any point in time at most F of these sensors may be producing erroneous results. To mitigate these faults the system determines the current conditions by creating a single interval obtained from the intersection of at least $N - F$ sensor readings. Such a system can then be configured to tolerate a predetermined number of faulty sensors. SLAT modifies this approach for use in secure localization by having non-beacon nodes select a value for F on the fly, such that their estimated positions are better in terms of both resiliency to compromised beacon nodes and certainty of position. Using a fixed value for F as is done in [9] requires one to guess what the maximum number of malicious nodes will be. Setting this value too low enables an adversary to cause the algorithm to not produce a result. Setting this value too high causes the algorithm to be too conservative, resulting in greater uncertainty in non-beacon position estimates.

III. PROBLEM STATEMENT

Given a sensor network composed of G randomly deployed beacon nodes equipped with GPS receivers and N randomly deployed non-beacon nodes, we are interested in utilizing the location information present on the beacon nodes to enable non-beacon nodes to estimate their positions. Specifically we are interested in a solution with the following characteristics: resiliency to attacks, resiliency to beacon node compromise, scalability, robustness to highly dynamic communications environments, and low resource requirements.

A. Node Assumptions

We assume that sensor nodes have the ability to dynamically scale their transmission power in order to increase their effective range [23], [24].

To make SLAT function regardless of the complexities of wireless communication (e.g. fading, interference, noise, etc.), we assume that each beacon node knows what its absolute maximum communication range, R , is. R should be determined such that in an actual implementation, it can be guaranteed that beacon messages will not be discernable outside of this range. To make calculations amenable to computationally “lightweight” sensor node hardware, we model this by a square of length $2R$ centered about each beacon node (see Figure 1). The reader should note that SLAT does not expect every node within a beacon node’s square to be capable of properly receiving its beacons. Rather,

it is expected that no node outside of a beacon's square will be capable of hearing the beacon's messages. In a real implementation we expect that a suitable value for R would be determined empirically.

We assume that beacon nodes are ordinary sensor nodes that are equipped with low-power GPS receivers. In this paper we shall assume that a node's position is given by an x and y coordinate, although this could be readily extended to three dimensions. We shall also assume that the error inherent to performing GPS positioning has a guaranteed upper limit. Crossbow Technology, Inc. offers the MTS420CA, an environmental monitoring board with GPS capabilities for use with their line of MOTE sensor nodes [11]. The MTS420CA utilizes Leadtek's GPS-9546 module [25]. Leadtek's GPS-9546 module is accurate to 1-10 m depending upon the corrective algorithm utilized. This unit can be configured for low-power operation so that the unit will only occasionally power on to receive satellite signals.

We also assume that beacon nodes are designed to be equipped with significantly more battery supplies than non-beacon nodes. Given that the current price of Crossbow's MTS420CA is \$375, we contend that the additional price to equip beacon nodes with additional battery supplies is negligible in comparison. The presence of additional battery supplies enables the beacon nodes to expend the additional power required to utilize their GPS units and increase their communication range. Thus if a non-beacon node's maximum communication range is r , then a beacon node's maximum communication range is $R > r$. By increasing the range of beacon nodes, non-beacon nodes are capable of receiving more beacon messages. This increases the system's robustness to compromised beacon nodes by increasing the number of legitimate beacon messages heard by non-beacon nodes.

B. Security Considerations

1) *Threat Model*: We assume an adversary that is capable of physically compromising sensor nodes, giving the adversary access to any keys, data, or program code located on the compromised node [2]–[7]. Given this assumption, secure localization solutions which rely on authentication alone to prevent attacks are not sufficient; a more robust solution must be resilient to beacon node compromise.

The adversary will perform two broad categories of location poisoning. In the first category the adversary attempts to disrupt localization of the entire network. In the second attack, the adversary is interested in disrupting the localization of a small region of nodes. This second category models situations where the adversary is interested in infiltrating a particular portion of the network, thus enabling the adversary to focus his/her attack upon a small region of the network (e.g. only inserting malicious nodes into a small area).

2) *Sensor Node Security Facilities*: We assume each sensor node will utilize the LEAP sensor network key management protocol [8]. In LEAP, each node is given a unique 4 byte integer which is used to uniquely identify it. Each node, u , derives its own master key, K_u , based upon information given to it by a trusted basestation. Using their respective master keys, each pair of neighboring nodes can independently negotiate a shared key simply by sharing their respective IDs. LEAP also makes provisions for allowing two nodes that are multiple hops away to setup shared keys. LEAP also enables nodes to create cluster keys, which enable secure broadcast communication. To create a cluster key, a node randomly selects a new key and uses its previously established pairwise keys to securely transmit the new key to its neighbors. The cluster key allows a node to use a single key to securely broadcast messages to all of its neighbors.

IV. SLAT: SECURE LOCALIZATION WITH ATTACK TOLERANCE

SLAT consists of two major components: a localization component (which is inherently robust to compromised beacon nodes) and an authentication component. We shall discuss these pieces of SLAT as separate entities.

A. Overview of Localization Component of SLAT

To explain localization in SLAT, consider the example in Figure 2; to ease exposition, we omit SLAT's security mechanisms and malicious nodes for this discussion. In this figure there are three beacon nodes labeled B1, B2, and B3 and nine non-beacon nodes labeled 1 through 9. In Figure 2a, each beacon node is surrounded by a square representing the absolute maximum range, R , of each beacon node. Any nodes outside of a beacon's square cannot receive its beacon messages. Note that because a beacon's actual communication range, R' , will be less than its maximum range, not all nodes within a beacon's square are guaranteed to receive its messages. (Also notice that since we assume asymmetric communication, a non-beacon node may not be capable of transmitting messages to a beacon that it receives messages from). Figure 2b indicates which beacons are correctly received by each non-beacon node. Also note that SLAT does not guarantee that all non-beacon nodes will end up with unique position estimates. Because nodes 1 and 5 receive the exact same beacons, their estimated positions will be the same.

Figure 2c illustrates the relationship between a beacon node's maximum communication range R , its actual communication range R' , its GPS error E_g , and its beacon messages. Notice the two points located in the center of this figure. The point labeled (x_{B1}, y_{B1}) corresponds to the beacon nodes actual position. The one labeled (x'_{B1}, y'_{B1}) corresponds to the position that the node believes it is at. The discrepancy between these two points is a result of the error resulting from using GPS. Because this error is guaranteed to be less than E_g , the beacon node knows that its actual position must reside within $(\pm E_g, \pm E_g)$ of its perceived location. This is illustrated by the inner box in the figure.

Now consider the discrepancy between R and R' . R represents the absolute maximum communication range of a beacon node. Under no circumstances is it possible for R' to exceed R . A beacon node whose actual position is (x_{B1}, y_{B1}) and whose actual communication range is R' will be correctly heard by any node within R' units its actual position. Thus any node within the dotted circle pictured in Figure 2c will be capable of hearing this beacon node.

Now consider the outer box in this Figure 2c. Recall that this beacon node knows that its actual position could reside anywhere within the inner box. Thus any node hearing the beacon node must be within R units of this inner box. This fact is depicted by the larger box in the figure. The coordinate of this box, which we denote as $(x_l, y_l), (x_u, y_u)$, are the coordinates included the beacon node's beacon message.

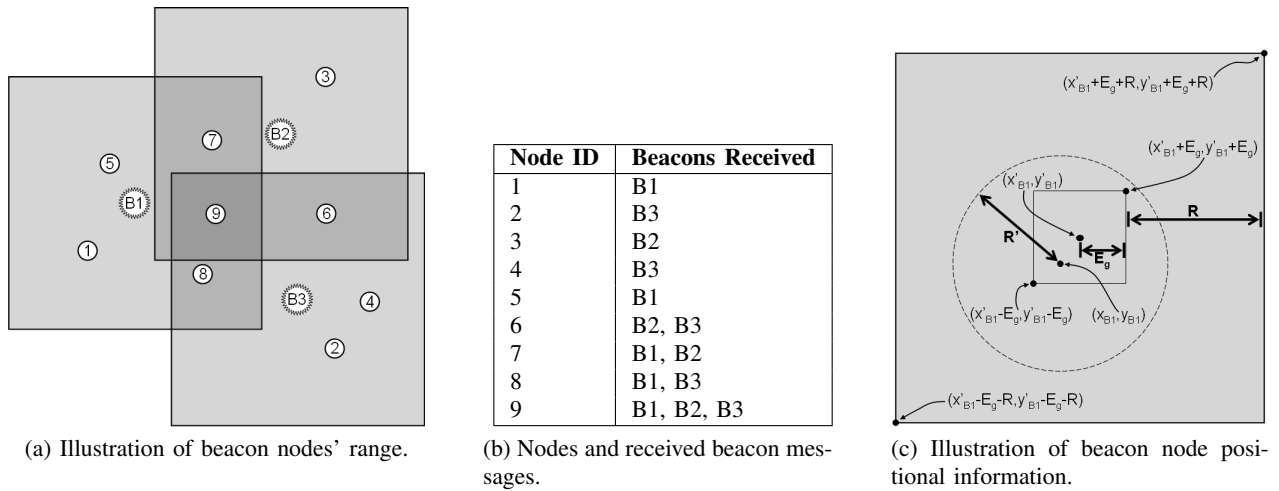


Fig. 2: Localization example showing 3 beacon nodes and 9 non-beacon nodes.

Now consider how SLAT would estimate the position of non-beacon node 9, which has received a beacon message from B1, B2, and B3. Let the three beacon nodes give their positional information as in Table I. Because node 9 can hear the beacon messages from B1, B2, and B3, it knows that it must reside in a location where these three intervals intersect. As will be explained further in Section IV-B, this enables node 9 to estimate its positional as $(22, 40), (30, 47)$.

TABLE I: Beacon nodes' positional information

Node ID	x_l	x_u	y_l	y_u
B1	10	30	35	55
B2	21	41	40	60
B3	22	42	27	47

The crucial requirements that must be ensured for this scheme to work are that the beacon nodes must know R and E_g . Knowing this information ensures that the beacon information that a non-beacon node obtains from legitimate beacon nodes will form intervals that intersect.

B. SLAT's Localization Algorithm

Before we introduce SLAT's localization algorithm, it is necessary to introduce some notation. First, we note that SLAT operates independently upon each coordinate in a beacon message. In the following explanation we explain how SLAT would operate upon a single coordinate, when in a real implementation SLAT would be operating upon two (i.e. the x and y direction) or possibly three coordinates (i.e. the $x, y,$ and z directions) at a time.

Let P represent the set of all position estimates received by a single non-beacon node (i.e. this set contains estimates from both malicious and nonmalicious nodes). Each position estimate is expressed as the interval $[LB_i, UB_i]$. Let LB and UB denote the set of all upper bound and lower bounds received by the non-beacon node respectively. Let LB_{Est} and UB_{Est} denote the non-beacon node's estimated lower and upper bounds upon completion of SLAT localization. Let $|A_i \geq B|$ denote the number of elements in some set B that are less than or equal to some number A_i . We also introduce two functions: min which returns the smaller of two numbers and max which returns the larger of two numbers.

Algorithm 1 describes localization in SLAT. This algorithm operates by separately determining the upper and lower bounds of a non-beacon node's position. SLAT initially assumes that there are no faulty beacon messages (line 3) and will iteratively increase the assumed number of faulty messages (line 13) until it can determine a resultant position estimate $[LB_{Est}, UB_{Est}]$. In this sense SLAT is an intrusion detection algorithm, which will only ignore as many beacon messages as is necessary to allow the non-beacon node to estimate its position. This iterative approach is appealing since it maximizes the number of beacon messages used to form a non-beacon node's position estimate while minimizing the width of the position estimate, thereby increasing the system's robustness to attack while decreasing the uncertainty in the non-beacon node's position. To best understand how SLAT achieves this two goals it is useful to consider SLAT as an optimization problem. On one hand we would like to have position estimates that are as tight as possible, implemented as $|LB_i \geq LB|$ and $|UB_i \leq UB|$. On the other hand we would like the position estimates to be formed from as many overlapping beacons as possible. Equivalently we would like to minimize the amount of non-overlap, implemented as $|LB_i \geq UB|$ and $|UB_i \leq LB|$. Combining both metrics, as done in lines 6 and 9, results in the desired resultant position estimates.

Algorithm 1 SLAT Localization

Require: $LB_j < UB_j$ for every $j \in P$

```

1:  $LB_{Est} \leftarrow \infty$ 
2:  $UB_{Est} \leftarrow -\infty$ 
3:  $F \leftarrow 0$ 
4: repeat
5:   for  $i = 1$  to  $|P|$  do
6:     if  $|LB_i \geq LB| - |LB_i \geq UB| \geq |P| - F$  then
7:        $LB_{Est} \leftarrow min(LB_{Est}, LB_i)$ 
8:     end if
9:     if  $|UB_i \leq UB| - |UB_i \leq LB| \geq |P| - F$  then
10:       $UB_{Est} \leftarrow max(UB_{Est}, UB_i)$ 
11:    end if
12:  end for
13:   $F \leftarrow F + 1$ 
14: until  $(UB_{Est} \neq -\infty$  AND  $LB_{Est} \neq \infty)$ 

```

Now let us consider how SLAT determines LB_{Est} (the process for determining UB_{Est} proceeds similarly). Figure 3a illustrates a non-beacon node (not pictured) that is estimating its position utilizing the five beacon messages labeled B1 through B5. Each beacon is illustrated with a line and two arrows. The numbers above the beacon's left and right arrows are the beacon's lower and upper bounds. Notice that in this figure that the maximum number of overlapping beacons is 3 (B1, B2, and B3). Consequently, SLAT will iterate through $F \leftarrow 0$ and $F \leftarrow 1$ without producing a resultant position estimate. Now consider how the algorithm would proceed once $F \leftarrow 2$. Specifically, consider the execution of line 6 in Algorithm 1. The results of executing line 6 for all possible values of LB_i is shown in Figure 3b. Given that $LB_i \leftarrow 377$ is the only value that satisfies the inequality in line 6, SLAT will select $LB_{est} \leftarrow 377$. Similarly SLAT will select $UB_{est} \leftarrow 428$. These two points are illustrated in Figure 3a by the two dotted lines. Upon visual inspection of this figure it should be clear to the reader that this interval spans the largest number of beacons.

At this point it may not be clear to the reader what purpose the min function in line 7 and the max function in line 10 of Algorithm 1 serve. These two functions deal with the case when the maximum amount of beacon overlap can be attained with multiple different values of LB_i . An example of such a case is illustrated in Figure 4. In this case a single non-beacon node is estimating its position with three non-intersecting beacons. Further, the maximum number of beacons that overlap is one. Thus all three beacons satisfy the inequality in line 6. To resolve this, SLAT takes the smallest candidate LB_i and the largest candidate UB_i , resulting in a position estimate will span any such

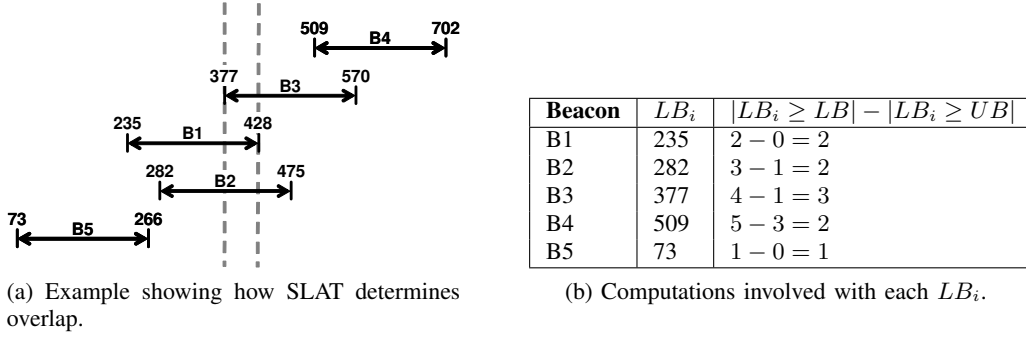


Fig. 3: Illustration of how SLAT computes LB_i .

points. Thus in this example, the non-beacon node determines that $LB_{est} \Leftarrow 10$ and $UB_{est} \Leftarrow 792$.

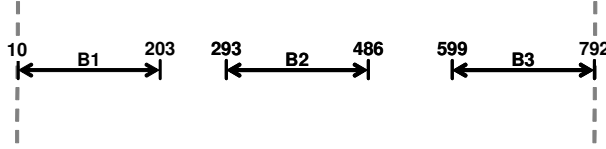


Fig. 4: Example illustrating how SLAT overcomes multiple inconsistent intervals.

C. Beacon Authentication

Now that we have introduced the localization aspect of SLAT, we will now introduce how SLAT prevents the creation of unauthorized beacon information. We first introduce the beacon message format and then discuss how beacon information is authenticated. The format of a beacon message is as follows:

$$B_i, (x_l, y_l), (x_u, y_u), t, MAC(k_i^c, B_i, x_l|y_l|x_u|y_u|t).$$

B_i represents the unique identifier for beacon node i . In our scheme each node is given a unique node identifier (ID). The notion of a node identifier can be extended to indicate whether the node is a beacon or non-beacon node. This is implemented by partitioning the ID address space. In Table II the four most significant bits of a node's ID indicate whether a node is a beacon node or a non-beacon node. This approach can be readily modified to alter the relative address space allotment between beacon and non-beacon nodes. Non-beacon nodes can verify that they are communicating with a beacon node by examining its ID. And because a node's ID is used to generate pairwise keys, any node claiming to be a beacon must have both a valid beacon ID and the master secret corresponding to that ID in order to properly establish pairwise keys with other nodes.

TABLE II: Sensor Node ID Address Space

Address Range	Node Type
00000000 to 0FFFFFFF	Beacon Node
10000000 to FFFFFFFF	Non-beacon Node

$(x_l, y_l), (x_u, y_u)$ denotes the beacon's positional information. Upper and lower bounds on both the x and y directions are given.

t denotes the beacon's monotonically increasing sequence number. The purpose of t is to prevent an adversary from replaying old beacon messages from the same beacon node, possibly when the beacon node was at a different location. Non-beacon nodes should keep track of the latest value for t associated with each beacon node. Non-beacon nodes should ignore beacon messages that contain a value for t that is less than the value for t that is currently being stored.

Beacon nodes create cluster keys, k_i^c , in order to authenticate their beacon messages; this is implemented by including a MAC (created using the cluster key) in each beacon message. Because beacon nodes have a significantly larger

communication range than non-beacon nodes, setting up pairwise keys (which are a prerequisite for establishing a cluster key) between some of the non-beacon nodes and the beacon nodes will require usage of multi-hop pairwise keys.

Authenticating beacon messages with cluster keys represents a compromise between communication overhead and security. In our approach an adversary could attain a particular beacon node’s cluster key by compromising either the beacon node or any of the non-beacon nodes with a copy of the cluster key. Further, compromise of a single non-beacon node will give the adversary access to any of the cluster keys located on that node. This could enable a malicious node to launch a Sybil attack [1], where a single node impersonates multiple nodes for the purpose of increasing the amount of damage they can cause. To help reduce the risk of such an attack, beacon nodes should immediately change their cluster keys upon determination that their key has been compromised. Such a protocol is orthogonal to our work; however, such compromises could be detected by beacon nodes observing beacon messages that they did not generate, but have been signed with their cluster key. Further, the fact that only the beacon node has access to all of the pairwise keys used to generate the last cluster key could be leveraged in order to securely create a new cluster key. Notice, that an adversary node cannot simply create its own identities for the purpose of launching a Sybil attack; rather, it must perform the more difficult process of attaining actual cluster keys by compromising nodes. To limit Sybil attacks further, one could require that each beacon node send an individual beacon message to each non-beacon node using pairwise shared keys. With such a design an adversary can only attain a single new identity by compromising a beacon node. However, this improved security greatly increases the communication overhead for the beacon node.

Wormhole attacks [26], in which a malicious node records messages from one location and replays them in a different location, are not a significant threat in SLAT. This is because recipient nodes will not have the cluster key needed to authenticate the MAC in the replayed message. Thus non-beacon nodes can simply ignore these messages.

D. Resilience to Compromised Beacon Nodes

To verify SLAT’s resilience to compromised beacon nodes, we have evaluated its ability to tolerate a number of different malicious node behaviors. The discussion of attacks in this section is from the perspective of a single non-beacon node receiving beacons from legitimate and malicious beacon nodes. Further, it is assumed that the range of beacon nodes, R , is 30 m. Each subsequent attack represents a higher level of complexity and a higher level of damage to victim nodes. The reader should note that for the reasons mentioned in Section IV-B, we will only focus on how SLAT performs for a single coordinate of interest. Further, we shall assume that malicious nodes generate position estimates that do not differ widely from those of legitimate beacon nodes. It is our assumption that non-beacon nodes will ignore beacon messages that differ greatly from the majority of received beacons.

1) *Arbitrarily Large Position Estimates*: The first malicious node behavior investigated was when malicious nodes produce position intervals that are arbitrarily large. More exactly, the malicious nodes send out beacons containing the interval $[-x \cdot 2R, x \cdot 2R]$ for some constant $x > 0$. This behavior illustrates that our technique is capable of filtering out largely inaccurate position estimates. An example of this behavior is illustrated in Figure 5. In Figure 5a there are ten nonmalicious beacon nodes and no malicious nodes present; in Figure 5b four malicious nodes have been added. The endpoints of each horizontal line correspond to the upper and lower position bounds provided by a single beacon message. The x -axis illustrates the relative distance between the beacon information and the non-beacon node whose actual position is denoted as 0. For instance beacon 3’s lower bound is 60 m less than the non-beacon node’s actual position. The y -axis denotes the ID of each beacon node; the malicious nodes are tagged with an “M.” The vertical dotted lines represent what the non-beacon node estimates its position as. It should be clear from comparing Figures 5b and 5a that the malicious beacon messages overlap with any nonmalicious beacon messages. Thus, so long as there is a single nonmalicious beacon node present, the non-beacon node will not be affected by this behavior.

2) *Random Position Estimates*: A second malicious node behavior would be to have each malicious node generate two random numbers r_1, r_2 in the interval $[-2R, 2R]$. Each malicious node could then indicate that its position interval is $[\min(r_1, r_2), \max(r_1, r_2)]$. Thus this behavior could be easily implemented by an adversary. An example of this behavior is illustrated in Figure 6, where ten legitimate beacon nodes are present in Figure 6a and in Figure 6b there are additionally four malicious nodes. In Figure 6b SLAT assumes that there are two faulty beacons: node M2 and either node M4 or node 10. Since both node M4 and node 10 would produce a maximal number of overlapping nodes (i.e. 12 nodes), SLAT creates a position estimate which spans both candidate beacons, as explained at the end of Section IV-B.

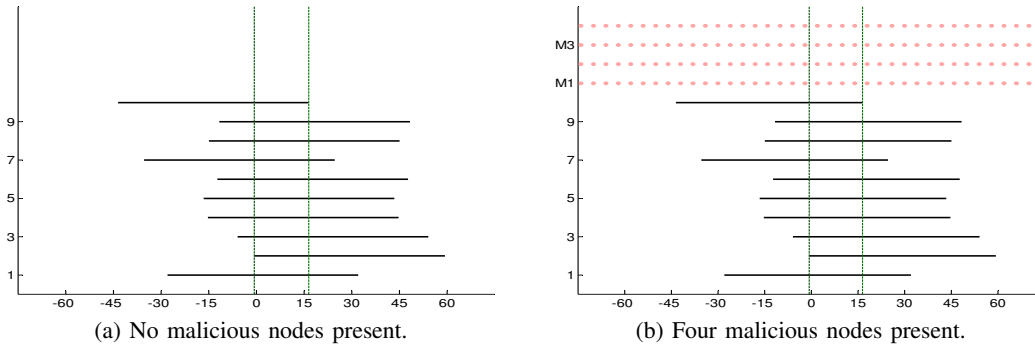


Fig. 5: Example of arbitrarily large position estimate malicious behavior.

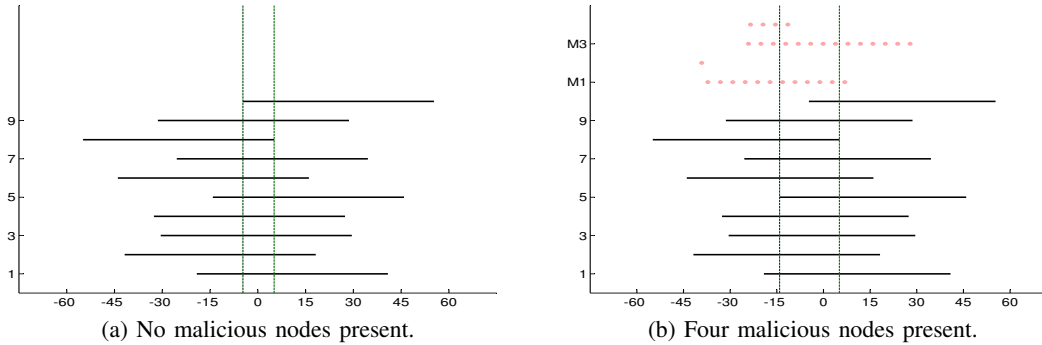


Fig. 6: Example of random position estimates malicious behavior.

We have used simulations to quantify the impact of malicious node behavior upon a single non-beacon node's position estimate. Figure 7 shows a particular example of the random position estimate behavior where there are 10 nonmalicious beacon nodes ($G = 10$) and the number of malicious beacon nodes, M , is varied between 0, 1, 2, 4, and 8. Each subfigure represents executing a simulator 130000 times, where each time the beacon nodes are given random positions with respect to the non-beacon node. For each simulation run, the non-beacon node's upper and lower position estimates were calculated. We then calculated the difference between the non-beacon node's actual position and both its upper and lower bound. These numbers were used as a measure of the error in the non-beacon node's estimated position. These error values were then used to derive the histograms in Figure 7. Consider position 0 along the x -axis of Figure 7a. The graph indicates that the relative frequency for this bar is about .4. This can be interpreted as meaning that out of 130000 simulation runs a total of $.4 \times 130000 = 52000$ simulation runs resulted in error values within the range of $(-3, 0]$. Alternatively, .4 can be interpreted as meaning the likelihood that a given simulation run will result in an error value within $(-3, 0]$. Note that the histograms for the upper bounds are omitted since they simply mirror the histograms for the lower bounds. The reader should note that the effects of the random estimates behavior are not dramatic.

3) *Shift Position Estimate*: A third malicious node behavior would be to have each malicious node listen to each of the position updates of the legitimate beacon nodes. The malicious nodes could then determine what upper and lower bounds the non-beacon node would come up with if there were no malicious nodes present. We assume that the malicious nodes could then determine whether the upper or the lower bound was further from the non-beacon node's actual position, and then they could fabricate their beacon messages so as to cause the non-beacon node's position estimate to be a very small interval around this upper/lower bound (i.e. it would appear to be nearly a single point). This behavior is illustrated in Figure 8. Notice that due to all 14 position estimates overlapping across such a small interval, the non-beacon node's position estimate looks as though it consists of a solitary single line; however, it is actually two lines separated by a small distance.

We have also relied upon simulations to evaluate the impact of the shift position behavior upon the non-beacon node's position estimate. The results of our experiments are illustrated in Figure 9. It should be clear from these figures that this attack does significantly more damage than the random position attack. However, the amount of error introduced by this attack is still quite small. Note that since each malicious node reports the same position estimate, there is no difference in the error induced by differing numbers of malicious nodes.

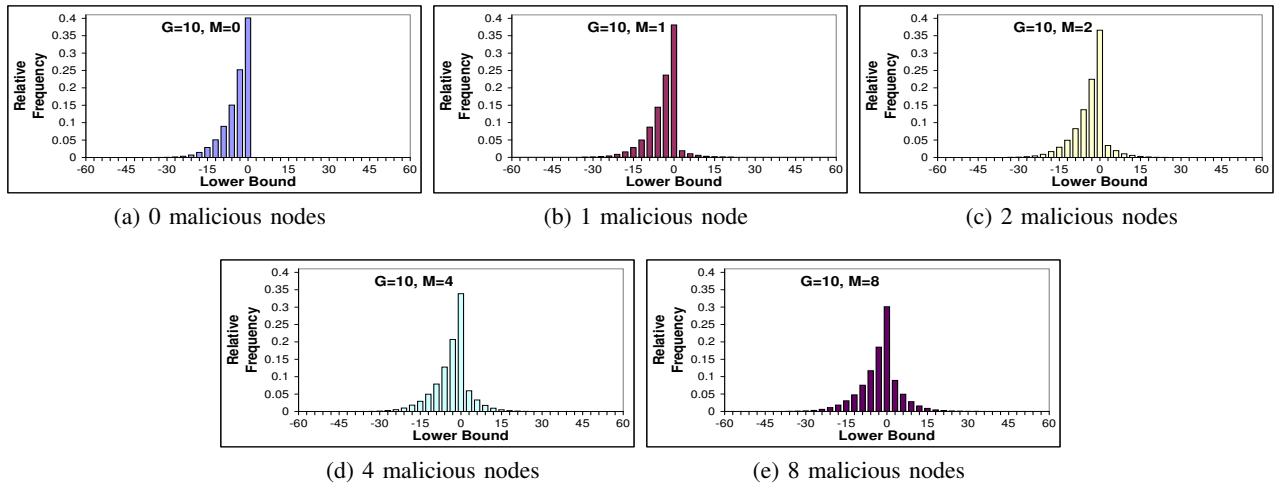


Fig. 7: Error caused by malicious nodes exhibiting random estimates behavior.

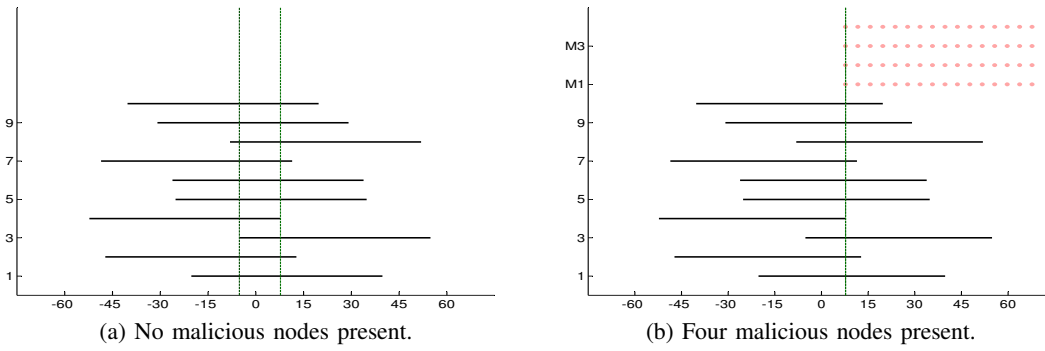


Fig. 8: Example of shift position estimates malicious behavior.

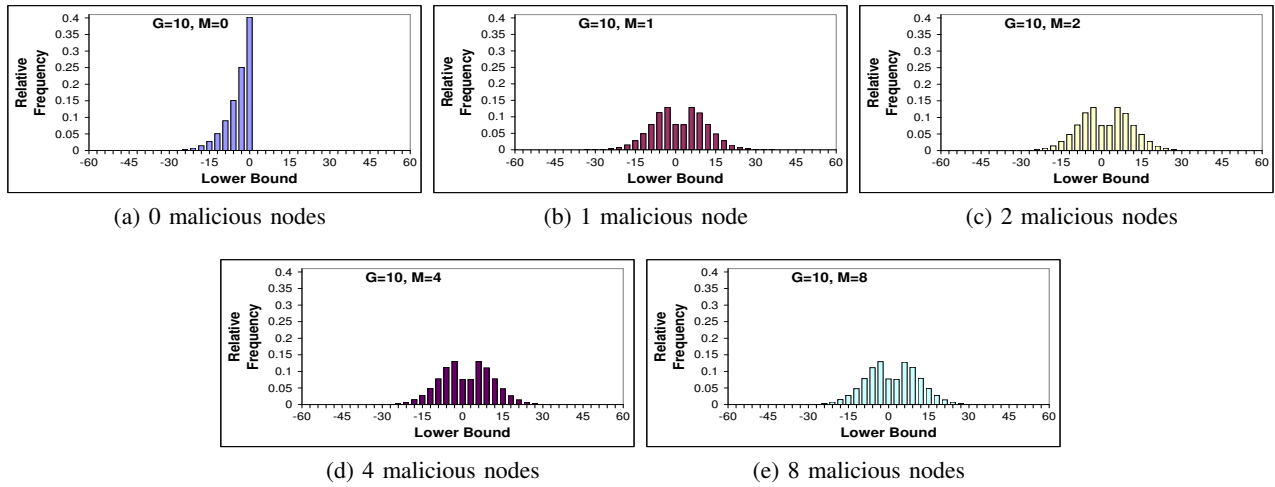


Fig. 9: Error caused by malicious nodes exhibiting shift position behavior.

4) *Override Position Estimate*: Our fourth malicious node behavior requires the malicious nodes to listen to the position estimates of legitimate beacon nodes and collude in an attempt to maximize their impact. This attack is based on the observation that SLAT will maximize the number beacons used to create a non-beacon node's position estimate. If there are G beacon nodes reporting position estimates, then under normal circumstances, the non-beacon node will estimate its position as the intersection of these G intervals. Now consider if there were also two malicious beacon nodes reporting position estimates. These two nodes could fabricate their position estimates so as to intersect

with $G - 1$ of the legitimate beacon node's position estimates. This would cause the non-beacon node to take its position as the intersection of the $G - 1$ legitimate nodes and the 2 malicious nodes. This could be extended such that M malicious nodes can collude to cause the non-beacon node to ignore $M - 1$ legitimate beacon nodes. Thus each additional malicious node could cause the non-beacon node's position estimate to shift further and further away from the non-beacon node's actual position. This behavior is represented pictorially in Figure 10. In this example there are four malicious nodes and ten legitimate beacon nodes. Notice that the four malicious nodes have nullified the effects of beacon nodes 5, 6, and 1. Further, the malicious nodes fabricated their beacons so that they marginally intersect with node 7.

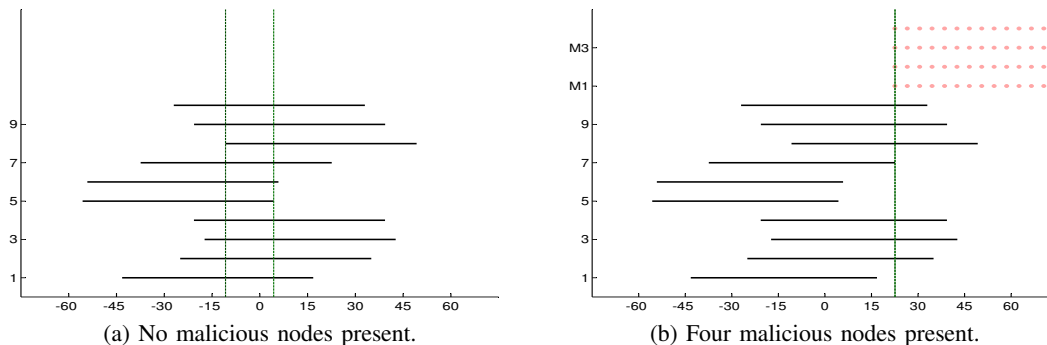


Fig. 10: Example of overrule position estimates malicious behavior.

We have also relied upon simulations to evaluate the impact of the overrule position behavior upon the non-beacon node's position estimate. It should be clear from this figure that the overrule position algorithm is the most successful attack that we have implemented to attack our location estimation algorithm. Analysis of this attack indicates that the degree to which a malicious location estimate affects a non-beacon node's position estimate is inversely proportional to its distance from the true location (as reported by the majority of properly operating beacons).

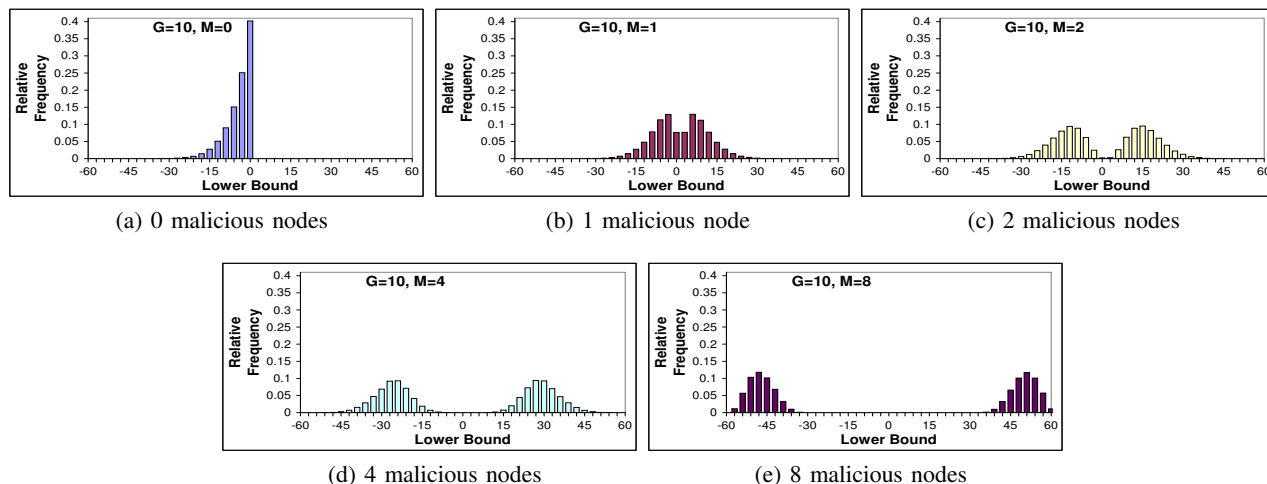


Fig. 11: Error caused by malicious nodes exhibiting overrule position estimates behavior.

E. Network-Wide Localization Error

In this section we quantify the impact of location poisoning and node compromise upon the network as a whole, rather than the localized attacks discussed earlier. To perform this analysis we have simulated an entire sensor network containing 200 beacon nodes, whose communication range is 150 m, and 1000 non-beacon nodes. All 1200 nodes are given random positions within a 1000 m by 1000 m square region. We have evaluated the network for the case where 0, 10, 20, and 40 random beacon nodes have been compromised and are now acting as malicious nodes. These malicious nodes utilize the random estimates behavior. Nodes using the shift position estimate and overrule position estimate behaviors must craft their beacon information for a specific target. Thus these attacks, by their very nature, cannot be readily used on a large scale.

We utilize two metrics to evaluate the impact of location poisoning on the network as a whole. The first metric measures the disparity between a node’s estimated position and its actual position. This is measured as the Manhattan distance from the center of a non-beacon node’s estimated position to its actual position. More formally this is represented as $xydiff = \frac{x_l + x_u}{2} - x_{actual} + \frac{y_l + y_u}{2} - y_{actual}$. From this figure we can see that the increase in error caused by the compromised nodes is quite small. For instance, the mean of average error when there are 0 compromised beacon nodes is about 2.5. The mean of average error only increases to about 5 when there are 40 compromised beacon nodes. The effect of compromised nodes in SLAT is particularly small in comparison to the nearly unbounded increase in the amount of error caused by malicious nodes in an unsecured localization scheme. Comparison of Figures 12b, 12c, and 12d indicates that a 100% increase in the number of compromised nodes only results in a 25% increase in the mean of non-beacon node average error.

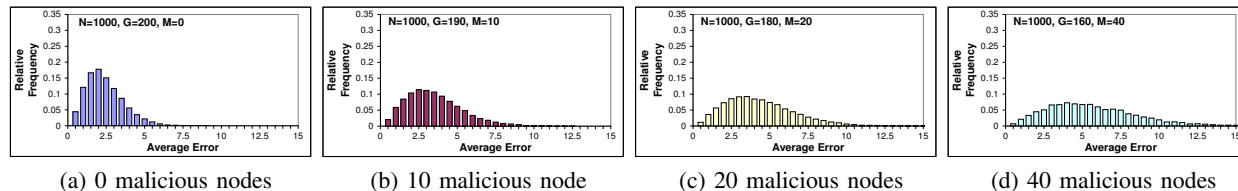


Fig. 12: Average width of non-beacon nodes estimated position for random estimates behavior.

Our second metric simultaneously measures the average width and height of all non-beacon nodes’ position estimates for a given network. More exactly, this metric determines the average value for $xywidth = (x_u - x_l)/2 + (y_u - y_l)/2$ for all 1000 non-beacon nodes in a given network. Thus this metric measures the uncertainty a non-beacon node has in its position. Figure 13 illustrates this metric. Each graph represents simulating 13000 different networks composed of randomly positioned nodes. From this figure we see that compromised nodes cause the non-beacon nodes to have narrower position estimates.

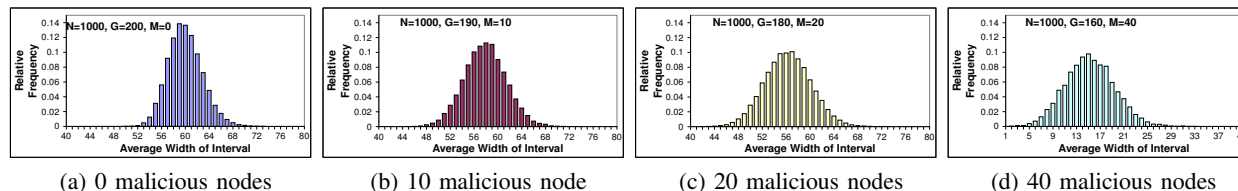


Fig. 13: Average width of non-beacon nodes estimated position for random estimates behavior.

We have found that an increased number of compromised beacon nodes causes a simultaneous increase in the error and perceived certainty of non-beacon node position. In other words non-beacon nodes have more faith in their position estimates, despite the fact that the error is higher. This indicates that localization approaches which have non-beacon nodes estimate their position based upon both beacon nodes and non-beacon nodes are particularly susceptible to location poisoning.

V. CONCLUSIONS

Accurate and secure localization is a necessity for the correct operation of many applications of sensor networks. In this paper we have proposed Secure Localization with Attack Tolerance (SLAT) to fulfill these requirements. SLAT contains two key components which make it resilient to attacks. First it requires that all beacon information originate from authenticated nodes. However, authentication alone is not sufficient to attain secure localization. Legitimate beacon nodes can be compromised, resulting in malicious but authenticated beacons. Thus the second key component to SLAT is resilience to compromised beacon nodes. To attain this second feature, non-beacon nodes in the SLAT algorithm estimate their positions based on the intersection of several beacon nodes in close proximity. Non-beacon nodes will perform intrusion detection to purge malicious beacon information from their position estimates.

We have utilized simulation to verify SLAT’s resilience towards attacks. We have evaluated SLAT against two broad categories of localization attacks, one where the adversary launches a focused attack upon a small region of the network and one where the adversary is interested in undermining the localization of the entire network. Our analysis

indicates that malicious beacon messages cannot differ widely from legitimate beacon messages, or else they will be detected as being malicious and be subsequently ignored; the best strategy from a malicious node's perspective, is to create beacon messages that are similar to the beacon messages from legitimate nodes. Consequently the effects of compromised beacon nodes in the SLAT algorithm are quite small. For example, we show that compromising 40 out of 200 beacon nodes in a simulated environment only increases the average location estimate error from 3 meters to 5 meters.

For future work we are considering implementing SLAT on actual sensor node hardware.

REFERENCES

- [1] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig, "The sybil attack in sensor networks: analysis & defenses," in *IPSN'04: Proceedings of the third international symposium on Information processing in sensor networks*. 2004, pp. 259–268, ACM Press.
- [2] David W. Carman, Peter S. Kruus, and Brian J. Matt, "Constraints and approaches for distributed sensor network security," Tech. Rep., NAI Labs #00-010, 2000.
- [3] David W. Carman, *Frontiers in Distributed Sensor Networks*, chapter Data Security Perspectives, CRC Press, 2004.
- [4] Haowen Chan and Adrian Perrig, "Security and privacy in sensor networks," *IEEE Computer Magazine*, pp. 103–105, 2003.
- [5] Anthony D. Wood and John A. Stankovic, "Denial of service in sensor networks," *Computer*, vol. 35, no. 10, pp. 54–62, 2002.
- [6] Adrian Perrig, John Stankovic, and David Wagner, "Security in wireless sensor networks," *Commun. ACM*, vol. 47, no. 6, pp. 53–57, 2004.
- [7] Chris Karlof and David Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.
- [8] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia, "Leap: efficient security mechanisms for large-scale distributed sensor networks," in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*. 2003, pp. 62–72, ACM Press.
- [9] Keith Marzullo, "Tolerating failures of continuous-valued sensors," *ACM Trans. Comput. Syst.*, vol. 8, no. 4, pp. 284–304, 1990.
- [10] Thales Navigations, "Specifications: Meridian color," http://www.magellangps.com/en/products/product_specs.asp?PRODID=171.
- [11] Crossbow Technology Inc, "Mts400/mts420 datasheet," http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MTS400-420_Datasheet.pdf.
- [12] Garmin, Ltd., "etrex specifications," <http://www.garmin.com/products/etrex/spec.html>.
- [13] Niveditha Sundaram and Parameswaran Ramanathan, "Connectivity based location estimation scheme for wireless ad hoc networks," in *Globecom*, 2002, vol. 1, pp. 143–147.
- [14] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*. 2003, pp. 81–95, ACM Press.
- [15] Nirupama Bulusu, Deborah Estrin, Lewis Girod, and John Heidemann, "Scalable coordination for wireless sensor networks: Self-configuring localization systems," in *Proceedings of the 6th IEEE International Symposium on Communication Theory and Application*. 2001, IEEE.
- [16] Koen Langendoen and Niels Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Computer Networks*, vol. 43, no. 4, pp. 499–518, 2003.
- [17] Andreas Savvides, Chih-Chieh Han, and Mani B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*. 2001, pp. 166–179, ACM Press.
- [18] Loukas Lazos and Radha Poovendran, "Serloc: secure range-independent localization for wireless sensor networks," in *WiSe '04: Proceedings of the 2004 ACM workshop on Wireless security*. 2004, pp. 21–30, ACM Press.
- [19] Saikat Ray, David Starobinski, Francesco De Pellegrini, Ari Trachtenberg, and Rachanee Ungrangsi, "Robust location detection in emergency sensor networks," in *IEEE INFOCOM 2003*, 2004, vol. 2, pp. 1044–1053.
- [20] Saikat Ray, David Starobinski, Ari Trachtenberg, and Rachanee Ungrangsi, "Robust location detection with sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 6, pp. 1016–1025, 2004.
- [21] Donggang Liu, Peng Ning, and Wenliang Du, "Attack-resistant location estimation in sensor networks," in *Proceedings of The Fourth International Symposium on Information Processing in Sensor Networks (IPSN '05)*, 2005, pp. 99–106.
- [22] Naveen Sastry, Umesh Shankar, and David Wagner, "Secure verification of location claims," in *WiSe '03: Proceedings of the 2003 ACM workshop on Wireless security*. 2003, pp. 1–10, ACM Press.
- [23] Rex Min and Anantha Chandrakasan, "A framework for energy-scalable communication in high-density wireless networks," in *ISLPED '02: Proceedings of the 2002 international symposium on Low power electronics and design*. 2002, pp. 36–41, ACM Press.
- [24] Eugene Shih, Seong-Hwan Cho, Nathan Ickes, Rex Min, Amit Sinha, Alice Wang, and Anantha Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*. 2001, pp. 272–287, ACM Press.
- [25] Leadtek Research Inc., "Leadtek gps module technical specification gps 9546," ftp://ftp1.leadtek.com/gps/9546/9546Manual_V1.12_20030806.pdf.
- [26] Yih-Chun Hu, Adrian Perrig, and David B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in *INFOCOM 2003*. 2003, vol. 3, pp. 1976 – 1986, IEEE Computer Society Press.