Attack-Resistant Location Estimation in Wireless Sensor Networks

Donggang Liu
The University of Texas at Arlington
Peng Ning, An Liu
North Carolina State University
Cliff Wang
Army Research Office
Wenliang Kevin Du
Syracuse University

Many sensor network applications require sensors' locations to function correctly. Despite the recent advances, location discovery for sensor networks in *hostile environments* has been mostly overlooked. Most of the existing localization protocols for sensor networks are vulnerable in hostile environments. The security of location discovery can certainly be enhanced by authentication. However, the possible node compromises and the fact that location determination uses certain physical features (e.g., received signal strength) of radio signals make authentication not as effective as in traditional security applications. This paper presents two methods to tolerate malicious attacks against range-based location discovery in sensor networks. The first method filters out malicious beacon signals on the basis of the "consistency" among multiple beacon signals, while the second method tolerates malicious beacon signals by adopting an iteratively refined voting scheme. Both methods can survive malicious attacks even if the attacks bypass authentication, provided that the benign beacon signals constitute the majority of the beacon signals. This paper also presents the implementation and experimental evaluation (through both field experiments and simulation) of all the secure and resilient location estimation schemes that can be used on the current generation of sensor platforms (e.g., MICA series of motes), including the techniques proposed in this paper, in a network of MICAz motes. The experimental results demonstrate the effectiveness of the proposed methods, and also give the secure and resilient location estimation scheme most suitalbe for the current generation of sensor networks.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms: Security, Design, Algorithms

Additional Key Words and Phrases: Sensor Networks, Security, Localization

This work is supported by the National Science Foundation (NSF) under grants CNS-0430223 and CNS-0430252. A. Liu and Wang's work is supported by the US Army Research Office (ARO) under Wang's staff research grant W911NF-04-D-0003-0001.

A preliminary version of this paper appeared in the *Proceedings of The Fourth International Symposium on Information Processing in Sensor Networks (IPSN '05), pages 99 – 106, April 2005*.

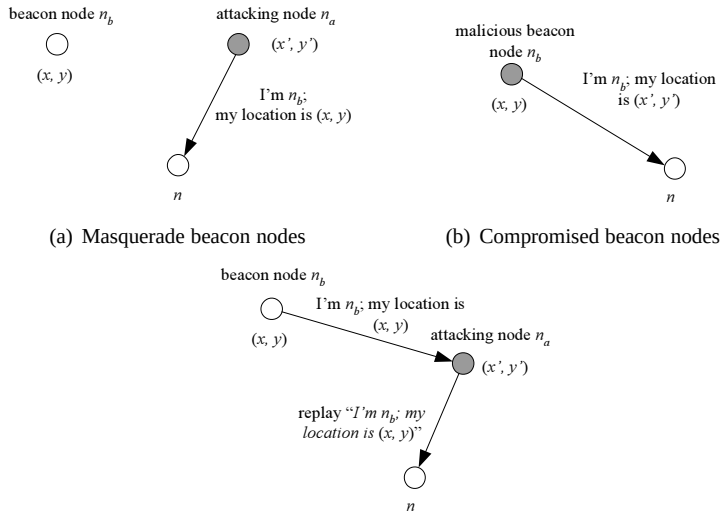
1. INTRODUCTION

Recent technological advances have made it possible to develop distributed sensor networks consisting of a large number of low-cost, low-power, and multi-functional sensor nodes that communicate in short distances through wireless links [Akyildiz et al. 2002]. Such sensor networks are ideal candidates for a wide range of applications such as health monitoring, data acquisition in hazardous environments, and military operations. The desirable features of distributed sensor networks have attracted many researchers to develop protocols and algorithms that can fulfill the requirements of these applications (e.g., [Perrig et al. 2001; Hill et al. 2000; Gay et al. 2003; Niculescu and Nath 2001; Intanagonwiwat et al. 2000; Newsome and Song 2003; Akyildiz et al. 2002]).

Sensors' locations play a critical role in many sensor network applications. Not only do applications such as environment monitoring and target tracking require sensors' location information to fulfill their tasks, but several fundamental techniques developed for wireless sensor networks also require sensor nodes' locations. For example, in geographical routing protocols (e.g., GPSR [Karp and Kung 2000] and GEAR [Yu et al. 2001]), sensor nodes make routing decisions at least partially based on their own and their neighbors' locations. As another example, in some data-centric storage applications such as GHT [Ratnasamy et al. 2002; Shenker et al. 2002], storage and retrieval of sensor data highly depend on sensors' locations. Indeed, many sensor network applications will not work without sensors' location information.

A number of location discovery protocols (e.g., [Savvides et al. 2001; Savvides et al. 2002; Niculescu and Nath 2003a; Nasipuri and Li 2002; Doherty et al. 2001; Bulusu et al. 2000; Niculescu and Nath 2003b; Nagpal et al. 2003; He et al. 2003]) have been proposed for wireless sensor networks in recent years. These protocols share a common feature: They all use some special nodes, called *beacon* (or *anchor*) *nodes*, which are assumed to know their own locations (e.g., through GPS receivers or manual configuration). These protocols work in two stages. In the first stage, non-beacon nodes receive radio signals called *beacon signals* from the beacon nodes. The packet carried by a beacon signal, which we call a *beacon packet*, usually includes the location of the beacon node. The non-beacon nodes then estimate certain measurements (e.g., distance between the beacon and the non-beacon nodes) based on features of the beacon signals (e.g., received signal strength indicator (RSSI), time difference of arrival (TDoA)). We refer to such a measurement and the location of the corresponding beacon node collectively as a *location reference*. In the second stage, a sensor node determines its own location when it has enough number of location references from different beacon nodes. A typical approach is to consider the location references as constraints that a sensor node's location must satisfy, and estimate it by finding a mathematical solution that satisfies these constraints with minimum estimation error. Existing approaches either employ *range-based* methods [Savvides et al. 2001; Savvides et al. 2002; Niculescu and Nath 2003a; Nasipuri and Li 2002; Doherty et al. 2001], which use the exact measurements obtained in stage one, or *range-free* ones [Bulusu et al. 2000; Niculescu and Nath 2003b; Nagpal et al. 2003; He et al. 2003; Lazos and Poovendran 2004], which only need the existences of beacon signals in stage one.

Despite the recent advances, location discovery for wireless sensor networks in *hostile environments*, where there may be malicious attacks, has been mostly overlooked. Many existing location discovery protocols become vulnerable in the presence of malicious attacks. As illustrated in Figure 1, an attacker may provide incorrect location reference by



(a) Masquerade beacon nodes (b) Compromised beacon nodes

(c) Replay attack

Fig. 1. Attacks against location discovery schemes

pretending to be valid beacon nodes (Figure 1(a)), compromising beacon nodes (Figure 1(b)), or replaying the beacon packets that he/she intercepted in different locations (Figure 1(c)). In either of the above cases, non-beacon nodes will determine their locations incorrectly. In either of these cases, non-beacon nodes will determine their locations incorrectly.

Without protection, an attacker may easily mislead the location estimation at sensor nodes and subvert the normal operation of sensor networks. The security of location discovery can certainly be enhanced by authentication. Specifically, each beacon packet should be authenticated with a cryptographic key only known to the sender and the intended receivers, and a non-beacon node accepts a beacon signal only when the beacon packet carried by the beacon signal can be authenticated. However, authentication does not guarantee the security of location discovery, either. An attacker may forge beacon packets with keys learned through compromised nodes, or replay beacon signals intercepted in different locations. Indeed, an attacker can introduce substantial location estimation errors by forging or replaying beacon packets. Thus, it is highly desirable to have additional methods to protect location discovery in sensor networks.

Several techniques has been developed recently to deal with the security problems of location discovery in wireless sensor networks [Sastry et al. 2003; Lazos and Poovendran 2004; Ray et al. 2003; Li et al. 2005; Capkun and Hubaux 2005; Lazos et al. 2005]. The location verification technique proposed in [Sastry et al. 2003] can be used to verify the relative distance between a verifying node and a sensor node. However, it does not provide a solution to conduct secure location estimation at non-beacon nodes. A robust location detection is developed in [Ray et al. 2003] using the idea of majority voting. However, it cannot be directly applied in resource constrained sensor networks due to its high computation and storage overheads. Similar to our attack-resistant MMSE techniques, a robust statistical method is independently discovered in [Li et al. 2005] to achieve robustness

through Least Median of Squares.

SeRLoc [Lazos and Poovendran 2004] protects location discovery with the help of sectored antennae at beacon nodes. Similar to the voting-based scheme proposed in this paper, SeRLoc can tolerate malicious attacks by adopting the idea of majority voting. SPINE [Capkun and Hubaux 2005] is developed to protect location discovery by using verifiable multilateration. However, the distance bounding techniques required for verifiable multilateration may not be available in sensor networks due to the difficulties to (1) deal with the external attacks in Ultrasound-based distance bounding and (2) achieve nanosecond processing and time measurements in Radio-based distance bounding. ROPE [Lazos et al. 2005] is developed by integrating SeRLoc and SPINE. However, it still requires nanosecond processing and time measurements that are not desirable for the current generation of sensor networks.

In this paper, we investigate two types of attack-resistant location estimation techniques to tolerate the malicious attacks against range-based location discovery in wireless sensor networks. The first technique, named *Attack-Resistant Minimum Mean Square Estimation (ARMMSE)*, is based on the observation that malicious location references introduced by attacks are intended to mislead a sensor node about its location, and thus are usually inconsistent with the benign ones. To exploit this observation, our methods identify malicious location references by examining the inconsistency among location references (indicated by the mean square error of estimation) and defeats malicious attacks by removing such malicious data.

We develop three variations of the ARMMSE method to identify malicious location references: *the brute-force ARMMSE algorithm*, *the greedy ARMMSE algorithm* and *the enhanced greedy ARMMSE algorithm*. The brute-force algorithm tries every combination of location references to identify the largest set of consistent location references. It introduces high computation overhead at sensor nodes. The greedy algorithm is developed to reduce the computation overhead. It works in rounds and removes the most suspicious location reference in each round. The enhanced greedy algorithm further improves the performance of the greedy algorithm by adopting a more efficient way to identify the most suspicious location reference. We also develop an algorithm to incrementally perform the MMSE computation in the enhanced greedy algorithm, which significantly reduces the computation cost. The end result is an efficient and resilient algorithm that can defend against malicious attacks aimed at location estimation.

Our second technique, a *voting-based location estimation* method, quantizes the deployment field into a grid of cells and has each location reference "vote" on the cells in which the node may reside. Moreover, we develop a method that allows iterative refinement of the "voting" results so that it can be executed in resource constrained sensor nodes.

We have implemented the proposed schemes on TinyOS [Hill et al. 2000], and evaluated the performance through both field experiments in a network of MICAz motes and simulation. To provide a realistic model in the simulation for the radio signal used for distance measurement, we perform an extensive set of experiments with MICAz motes to profile the channel characteristics. We compare all the attack-resistant schemes that can be implemented on the current sensor platforms (e.g., MICAz motes) through experimental evaluation, aiming at identifying the algorithm most suitable for the current generation of sensor networks. Our experiments indicate that (1) the investigated schemes (including both approaches proposed in this paper) can effectively remove the effect of malicious lo-

cation references when the majority of location references are benign, and (2) among all the algorithms that can be implemented on the current sensor platforms, the enhanced greedy ARMMSE algorithm has the least execution time while providing a similar resiliency to the other schemes.

The rest of the paper is organized as follows. Section 2 discusses assumptions and the threat model. Sections 3 and 4 present the ARMMSE location estimation and the voting-based location estimation techniques, respectively. Section 5 provides the security analysis for the proposed schemes. Section 6 discusses the evaluation methodology for our experimental evaluation and presents the detailed evaluation results. Section 7 discusses related work, and Section 8 concludes this paper.

2. ASSUMPTIONS AND THREAT MODEL

In this paper, we present two approaches to dealing with malicious attacks against location discovery in wireless sensor networks. The first approach is extended from the minimum mean square estimation (MMSE). It uses the mean square error as an indicator to identify and remove malicious location references. The second one adopts an iteratively refined voting scheme to tolerate malicious location references introduced by attackers.

Our techniques are purely based on a set of location references. The location references may come from beacon nodes that are either single hop or multiple hops away, or from those non-beacon nodes that already estimated their locations. We do not distinguish these location references, though the effect of "error propagation" may affect the performance of our techniques due to the estimation errors at non-beacon nodes. We consider such investigations as possible future work. Since our techniques only utilize the location references from beacon nodes, there is no extra communication overhead involved when compared to the previous localization schemes.

We assume all beacon nodes are uniquely identified. In other words, a non-beacon node can identify the original sender of each beacon packet based on the cryptographic key used to authenticate the packet. This can be easily achieved with a pairwise key establishment scheme [Eschenauer and Gligor 2002; Chan et al. 2003; Du et al. 2003] or a broadcast authentication scheme [Perrig et al. 2001].

We assume each non-beacon node uses at most one location reference derived from the beacon signals sent by each beacon node. As a result, even if a beacon node is compromised, the attacker that has access to the compromised key can only introduce at most one malicious location reference to a given non-beacon node by impersonating the compromised node.

For simplicity, we assume the distances measured from beacon signals (e.g., with RSSI or TDoA [Savvides et al. 2001]) are used for location estimation. (Our techniques can certainly be modified to accommodate other measurements such as angles.) For the sake of presentation, we denote a location reference obtained from a beacon signal as a triple $\langle x, y, \delta \rangle$, where (x, y) is the location of the beacon declared in the beacon packet, and δ is the distance measured from its beacon signal.

We assume an attacker may change any field in a location reference through, for example, compromised nodes or wormhole attacks [Hu et al. 2003]. In other words, it may declare a wrong location in its beacon packets, or carefully manipulate the beacon signals to affect the distance measurement by, for example, manipulating the signal strength when RSSI is used for distance measurement. We also assume multiple malicious beacon nodes

may collude together to make the malicious location references appear to be "consistent". Our techniques can still defeat such colluding attacks as long as the majority of location references are benign.

An attacker may also launch physical attacks to change sensors' location after they finish location estimation. (Indeed, other factors (e.g., wind) may also change sensors' locations.) Such threats cannot be directly addressed by attack-resistant location estimation. However, we assume each sensor can perform location estimation periodically based on fresh location references to mitigate such threats. An attacker may also jam the communication channel to, for example, prevent the successful transmission and receiving of beacon signals. This is a common threat to all wireless networks. However, we assume the attacker cannot jam the communication channel continuously without being detected and removed.

3. ATTACK-RESISTANT MINIMUM MEAN SQUARE ESTIMATION

Intuitively, a location reference introduced by a malicious attack is aimed at misleading a sensor node about its location. Thus, it is usually "different" from benign location references. When there are redundant location references, there must be some "inconsistency" between the malicious location references and the benign ones. (An attacker may still have a location reference consistent with the benign ones after changing both the location and the distance values. However, such a location reference will not generate significantly negative impact on location determination.) To take advantage of this observation, we propose to use the "inconsistency" among the location references to identify the malicious ones, and discard them before finally estimating the locations at sensor nodes.

In this paper, we assume a sensor node uses an MMSE-based method (e.g., [Savvides et al. 2001; Savvides et al. 2002; Niculescu and Nath 2003a; Nasipuri and Li 2002; Doherty et al. 2001; Niculescu and Nath 2003b]) to estimate its own location. Thus, most current range-based localization methods can be used with this technique. To harness this observation, we first estimate the sensor's location with the MMSE-based method and then assess if the estimated location could be derived from a set of consistent location references. If yes, we accept the estimation result; otherwise, we identify and remove the most "inconsistent" location reference, and repeat the above process. This process may continue until we find a set of consistent location references or it is not possible to find such a set.

3.1 Checking the Consistency of Location References

We use the mean square error ς^2 of the distance measurements based on the estimated location as an indicator of the degree of inconsistency, since all the MMSE-based methods estimate a sensor node's location by (approximately) minimizing this mean square error. Other indicators are possible but need further investigation.

DEFINITION 1. *Given a set of location references* $\mathcal{L} = \{\langle x_1, y_1, \delta_1\rangle, \langle x_2, y_2, \delta_2\rangle, ..., \langle x_m, y_m, \delta_m\rangle\}$ *and a location* $(\hat{x}, \hat{y})$ *estimated based on* $\mathcal{L}$*, the* mean square error of this location estimation *is*

$$\varsigma^2 = \sum_{i=1}^{m} \frac{(\delta_i - \sqrt{(\hat{x} - x_i)^2 + (\hat{y} - y_i)^2})^2}{m}.$$

Intuitively, the more inconsistent a set of location references is, the greater the corresponding mean square error should be. To gain further understanding, we performed an experiment through simulation with the MMSE-based method in [Savvides et al. 2001].

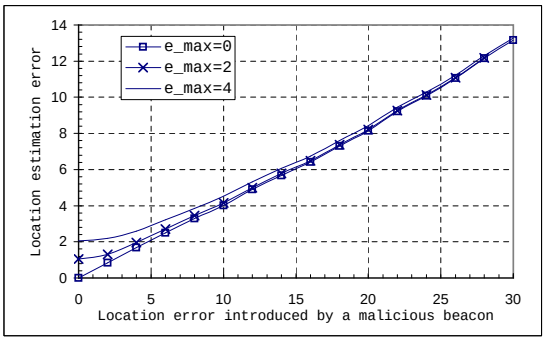


Fig. 2. Location estimation error. Unit of measurement for x and y axes: meter

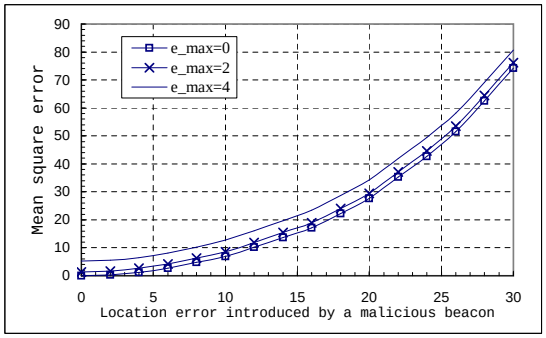


Fig. 3. Mean square error ς^2. Unit of measurement for x-axis: meter

We assume the distance measurement error is uniformly distributed between $-e_{max}$ and e_{max}. We used 9 honest beacon nodes and 1 malicious beacon node evenly deployed in a $30m \times 30m$ field. The node that estimates location is positioned at the center of the field. The malicious beacon node always declares a false location that is x meters away from its real location, where x is a parameter in our experiment.

Figures 2 and 3 show the location estimation error (i.e., the distance between a sensor's real location and the estimated location) and the mean square error ς^2 when x increases. As these figures show, if a malicious beacon node increases the location estimation error by introducing greater errors, it also increases the mean square error ς^2 at the same time. This further demonstrates that the mean square error ς^2 is potentially a good indicator of inconsistent location references.

In this paper, we choose a simple, threshold-based method to determine if a set of location references is consistent. Specifically, a set of location references $\mathcal{L} = \{\langle x_1, y_1, \delta_1\rangle, \langle x_2, y_2, \delta_2\rangle, ..., \langle x_m, y_m, \delta_m\rangle\}$ obtained at a sensor node is *τ-consistent w.r.t. an MMSE-based method* if the method gives an estimated location $(\hat{x}, \hat{y})$ such that the mean square error of this location estimation

$$\varsigma^2 = \sum_{i=1}^{m} \frac{(\delta_i - \sqrt{(\hat{x} - x_i)^2 + (\hat{y} - y_i)^2})^2}{m} \leq \tau^2.$$

3.2 Determining Threshold τ

The determination of threshold τ depends on the measurement error model, which can be established before network deployment by, for example, conducting field experiments. We assume that the threshold is stored on each sensor node. Usually, the movement of sensor nodes (beacon or non-beacon nodes) does not have significant impact on this threshold, since the measurement error model will not change significantly in most cases. Moreover, since such a model depends on the physical features of radio signals and the environments, it is also difficult for an attacker to manipulate. Hence, we assume such model is available. However, when the error model changes frequently and significantly, the performance of our techniques may be affected. In this paper, we assume the measurement error model will not change.

Note that the malicious beacon signals usually increase the variance of estimation. Thus, having a lower bound (e.g., Cramer-Rao bound) is not enough for us to filter malicious beacon signals. In fact, the upper bound or the distribution of the mean square error are more desirable. In this paper, we study the distribution of the mean square error ς^2 when there are no malicious attacks, and use this information to help determine the threshold τ.

Since there is no other error besides the distance measurement error, a benign location reference $\langle x_i, y_i, \delta \rangle$ obtained by a sensor node at (x, y) must satisfy:

$$|\delta - \sqrt{(x - x_i)^2 + (y - y_i)^2}| \leq \epsilon,$$

where ϵ is the maximum distance measurement error.

All the localization techniques are aimed at estimating a location as close to the sensor's real location as possible. Thus, we may assume the estimated location is very close to the real location when there are no attacks. Next, we derive the distribution of the mean square error ς^2 using the real location as the estimated location, and compare it with the distribution obtained through simulation when there are location estimation errors.

The measurement error of a benign location reference $\langle x_i, y_i, \delta_i \rangle$ can be computed as $e_i = \delta_i - \sqrt{(x - x_i)^2 + (y - y_i)^2}$, where (x, y) is the real location of the sensor node. Assuming the measurement errors introduced by different benign location references are independent, we can get the distribution of the mean square error through the following Lemma.

LEMMA 1. *Let $\{e_1, ..., e_m\}$ be a set of independent random variables, and μ_i, σ_i^2 be the mean and the variance of e_i^2, respectively. If the estimated location of a sensor node is its real location, the probability distribution of ς^2 is*

$$\lim_{m \to \infty} F[\varsigma^2 \leq \varsigma_0^2] = \Phi(\frac{m\varsigma_0^2 - \mu'}{\sigma'}),$$

where $\mu' = \sum_{i=1}^{m} \mu_i$, $\sigma' = \sqrt{\sum_{i=0}^{m} \sigma_i^2}$, and $\Phi(x)$ is the probability of a standard normal random variable being less than x.

PROOF. Obviously, the mean square error can be computed by $\varsigma^2 = \sum_{i=1}^{m} \frac{e_i^2}{m}$. Thus, the cumulative distribution function can be calculated by

$$F(\varsigma^2 \leq \varsigma_0^2) = F(\sum_{i=1}^{m} e_i^2 \leq m\varsigma_0^2).$$

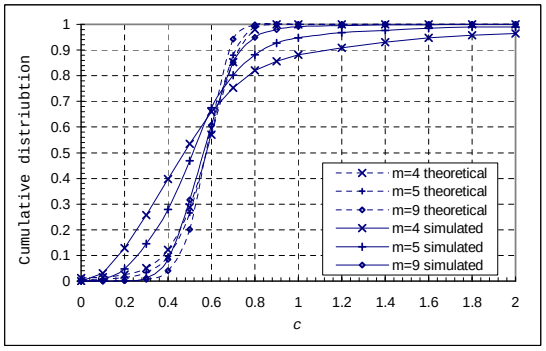


Fig. 4. Cumulative distribution function $F(\varsigma^2 \leq \varsigma_0^2)$. Let $c = \frac{\varsigma_0}{\epsilon}$.

Since $\{e_1^2, e_2^2, \cdots, e_m^2\}$ are independent, according to the central limit theorem, we have

$$\lim_{m\to\infty} P(\frac{S_m - \mu'}{\sigma'} \leq x) = \Phi(x),$$

where $S_m = \sum_{i=1}^{m} e_i^2$. Thus, we have

$$\begin{aligned} \lim_{m\to\infty} F(\varsigma^2 \leq \varsigma_0^2) &= \lim_{m\to\infty} F(S_m \leq m\varsigma_0^2) \\ &= \lim_{m\to\infty} P(\tfrac{S_m-\mu'}{\sigma'} \leq \tfrac{m\varsigma_0^2-\mu'}{\sigma'}) \\ &= \Phi(\tfrac{m\varsigma_0^2-\mu'}{\sigma'}) \end{aligned}$$

□

Lemma 1 describes the probability distribution of ς^2 based on a sensor's real location. Though it is different from the probability distribution of ς^2 based on a sensor's estimated location, it can be used to approximate such distribution in most cases.

Let us further assume a simple model for measurement errors, where the measurement error is evenly distributed between $-\epsilon$ and ϵ. Then the mean and the variance for e_i are 0 and $\frac{\epsilon^2}{3}$, respectively, and the mean and the variance for any e_i^2 are $\frac{\epsilon^2}{3}$ and $\frac{4\epsilon^4}{45}$, respectively. Let $c = \frac{\varsigma_0}{\epsilon}$, we have

$$F(\varsigma^2 \leq (c \times \epsilon)^2) = \Phi(\frac{\sqrt{5m}(3c^2-1)}{2}).$$

Figure 4 shows the probability distribution of ς^2 derived from Lemma 1 and the simulated results using sensors' estimated locations. We can see that when the number of location references m is large (e.g., $m = 9$) the theoretical result derived from Lemma 1 is very close to the simulation results. However, when m is small (e.g., $m = 4$), there are observable differences between the theoretical results and the simulation. The reasons are twofold. First, our theoretical analysis is based on the central limit theorem, which is only an approximation of the distribution when m is a large number. Second, we used the MMSE-based method proposed in [Savvides et al. 2001] in the simulation, which estimates a node's location by only *approximately* minimizing the mean square error. (Otherwise, the value of ς^2 for benign location references should never exceed ϵ^2.)

Figure 4 gives three hints about the choice of the threshold τ. First, when there are enough number of benign location references, a threshold less than the maximum mea-

surement error is enough. For example, when $m = 9$, $\tau = 0.8\epsilon$ can guarantee the nine benign location references are considered consistent with high probability. Besides, a large threshold may lead to the failure to filter out malicious location references. Second, when m is small (e.g. 4), the cumulative probability becomes flatter and flatter when $c > 0.8$. This means that setting a large threshold τ for small m may not help much to guarantee the consistency test for benign location references; instead, it may give an attacker high chance to survive the detection. Third, the threshold cannot be too small; otherwise, a set of benign location references has high probability to be determined as a non-consistent reference set.

Based on the above observations, we propose to choose the value for τ with a hybrid method. Specifically, when the number of location references is large (e.g., more than 8), we determine the value of τ based on Lemma 1. Specifically, we choose a value of τ corresponding to a high cumulative probability (e.g., 0.9). When the number location references is small, we perform simulation to derive the actual distribution of the mean square error, and then determine the value of τ accordingly. Since there are only a small number of simulations to run, we believe this approach is practical.

3.3 Identifying the Largest Consistent Set

Since the MMSE-based methods can deal with measurement errors better if there are more benign location references, we should keep as many benign location references as possible when the malicious ones are removed. This implies we should get the largest set of consistent location references.

3.3.1 *Brute-force Algorithm (BARMMSE).* Given a set $\mathcal{L}$ of n location references and a threshold τ, a simple approach to computing the largest set of τ-consistent location references is to check all subsets of $\mathcal{L}$ with i location references about τ-consistency, where i starts from n and drops until a subset of $\mathcal{L}$ is found to be τ-consistent or it is not possible to find such a set. Thus, if the largest set of consistent location references consists of m elements, a sensor node has to use the MMSE method at least $1 + \binom{n}{m+1} + \cdots + \binom{n}{n}$ times to find out the right one. If $n = 10$ and $m = 5$, a node needs to perform the MMSE method for at least 387 times. It is certainly not desirable to do such expensive operations on resource constrained sensor nodes.

3.3.2 *Greedy Algorithm (GARMMSE).* To reduce the computation on sensor nodes, we may use a greedy algorithm, which is simple but suboptimal. This greedy algorithm works in rounds. It starts with the set of all location references in the first round. In each round, it first verifies if the current set of location references is τ-consistent. If yes, the algorithm outputs the estimated location and stops. Optionally, it may also output the set of location references. Otherwise, it considers all subsets of location references with one fewer location reference, and chooses the subset with the least mean square error as the input to the next round. This algorithm continues until it finds a set of τ-consistent location references or when it is not possible to find such a set (i.e., there are only 3 remaining location references).

The greedy algorithm significantly reduces the computational overhead in sensor nodes. To continue the earlier example, a sensor node only needs to perform MMSE operations for about 50 times (instead of 387 times) using this algorithm. In general, a sensor node needs to use a MMSE-based method for at most $1+n+(n-1)+\cdots+4 = 1+\frac{(n-3)(n+4)}{2}$

times.

However, as we mentioned, the greedy algorithm cannot guarantee that it can always identify the largest consistent set. It is possible that benign location references are removed. In out earlier version of this paper [Liu et al. 2005a], we note that this generates a big impact on the accuracy of location estimation – especially when there are multiple malicious location references. To deal with this problem, we develop an enhanced greedy algorithm in the following. The new algorithm is based on an efficient approach to identifying the most suspicious location reference from a set of location references.

3.3.3 *Enhanced Greedy Algorithm (EARMMSE).* In the previous discussion, we only consider the consistency of 3 or more location references. A further investigation also reveals that two benign location references are usually "consistent" with each other in the sense that there exists at least one location in the deployment field on which both location references agree. Hence, when the majority of location references are benign, we can usually find many location references that are consistent with a benign location reference. In addition, when a malicious location reference tries to create a larger location error, the number of location references that are consistent with the malicious one will decrease quickly.

According to the above discussion, for each location reference, we simply count the number of location references that are consistent with this location reference. We call this number the *degree of consistency* and use it to rank the suspiciousness of the location references received at a particular non-beacon node. The smaller the degree is, the more likely that the corresponding location reference is malicious.

The consistency between two location references can be verified as follows. For any location reference $\langle x, y, \delta \rangle$, the non-beacon node derives the area that it may reside based on this location reference. This area can be represented by a ring centered at (x, y), with the inner radius $\max\{\delta - \epsilon, 0\}$ and the outer radius $\delta + \epsilon$, where ϵ is the maximum distance error. For the sake of presentation, we refer to such a ring a *candidate ring (centered) at location* (x, y). The non-beacon node then check whether the candidate rings of two location references overlap each other. If yes, they are consistent; otherwise, they are not consistent.

The algorithm to check whether the candidate rings of two location references $a = \langle x_a, y_a, \delta_a \rangle$ and $b = \langle x_b, y_b, \delta_b \rangle$ overlap can be done efficiently in the following way. Let d_{ab} denote the distance between (x_a, y_a) and (x_b, y_b). Let $rmax(x)$ and $rmin(x)$ denote the outer radius and the inner radius of the candidate ring of location reference x respectively. We can easily figure out that the candidate rings of location references a and b will not overlap when either of the following three conditions is true: (1) $d_{ab} > rmax(a) + rmax(b)$, (2) $d_{ab} + rmax(a) < rmin(b)$ and (3) $d_{ab} + rmax(b) < rmin(a)$.

Similar to the greedy algorithm, the enhanced algorithm to identify the largest consistent set starts with the set of all location references in the first round. In each round, it verifies whether the current set of location references is τ-consistent. If yes, the algorithm outputs the estimated location and stops. Optionally, it may also output the set of location references. If not, it removes the location reference corresponding to the smallest degree and use the remaining location references as the input to the next round. This algorithm continues until it finds a set of τ-consistent location references or when it is not possible to find such a set (i.e., there are only 3 remaining location references).

The enhanced algorithm not only improves the accuracy of location estimation in the

presence of malicious attacks, but also reduces the computation overhead significantly since it can identify the most suspicious location reference efficiently and effectively. To continue the earlier example, a non-beacon node only needs to perform MMSE operations for 5 times. In general, a non-beacon node needs to use a MMSE-based method for at most $n-3$ times.

3.4 Incremental Evaluation

ARMMSE uses the basic MMSE method in multiple rounds, with overlapping location references across rounds. In this section, we develop an efficient approach for performing multi-round MMSE operations in such situations. We call this approach *incremental MMSE*. It exploits the overlapping subsets of location references in different rounds to reduce unnecessary computation. We will see in our experimental evaluation that this approach significantly reduces the computation time (and thus the energy) required by ARMMSE.

In the following, we first give a brief introduction to the basic MMSE method proposed in [Savvides et al. 2001], and then discuss the incremental MMSE approach.

3.4.1 *Basic MMSE Method.* Assume a sensor node has obtained a set of m location references from the reachable beacon nodes, $\{(x_1, y_1, d_1), (x_2, y_2, d_2), \ldots, (x_m, y_m, d_m)\}$. Suppose the estimated location is $(\hat{x}, \hat{y})$. Thus, the error of the measured distance between the regular node and the ith ($1 \leq i \leq m$) beacon node can be expressed as the difference between the measured distance d_i and the estimated distance $d_i - \sqrt{(\hat{x}-x_i)^2 + (\hat{y}-y_i)^2}$.

The basic MMSE method obtains the location estimate $(\hat{x}, \hat{y})$ by minimizing the mean square error (MSE)

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left[\sqrt{(\hat{x}-x_i)^2 + (\hat{y}-y_i)^2} - d_i\right]^2.$$

To accommodate the limited computational power of current generation of sensor nodes (e.g., MICAz motes), the basic MMSE method uses an approximate approach to estimating the location $(\hat{x}, \hat{y})$ [Savvides et al. 2001]. Specifically, the location estimate $(\hat{x}, \hat{y})$ can be calculated using equation (1) as follows:

$$b = (X^T X)^{-1} X^T Y, \tag{1}$$

where

$$b = \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix},$$

$$X = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3) \\ \vdots & \vdots \\ 2(x_1 - x_m) & 2(y_1 - y_m) \end{bmatrix},$$

and

$$Y = \begin{bmatrix} x_1^2 + y_1^2 - d_1^2 - x_2^2 - y_2^2 + d_2^2 \\ x_1^2 + y_1^2 - d_1^2 - x_3^2 - y_3^2 + d_3^2 \\ \vdots \\ x_1^2 + y_1^2 - d_1^2 - x_m^2 - y_m^2 + d_m^2 \end{bmatrix}.$$

3.4.2 *Incremental ARMMSE.* The key to incremental ARMMSE is the careful arrangement of the input of an earlier MMSE estimation to be a subset of input of a later MMSE estimation. Given the partial results from a previous round, we only need to perform the computation related to the newly added data by reusing the intermediate results in the earlier calculation.

Suppose that the basic MMSE method has been performed on m location references, and that all intermediate matrices have been saved. For a newly added location reference $(x_{m+1}, y_{m+1}, d_{m+1})$, we can reuse all previous calculations, and only run a few new calculations to complete the new MMSE estimation on the $m+1$ location reference.

Consider equation (2), which is a part of the basic MMSE computation. (See Section 3.4.1.) The new computation is $x_{m+1}^2 + y_{m+1}^2 - d_{m+1}^2$, which is highlighted in equation (2). Similarly, we highlight all the new calculations (in boxed frame) in equations (2-8).

$$Y = \begin{bmatrix} x_1^2 + y_1^2 - d_1^2 - x_2^2 - y_2^2 + d_2^2 \\ x_1^2 + y_1^2 - d_1^2 - x_3^2 - y_3^2 + d_3^2 \\ \vdots \\ x_1^2 + y_1^2 - d_1^2 - x_m^2 - y_m^2 + d_m^2 \\ x_1^2 + y_1^2 - d_1^2 - \boxed{(x_{m+1}^2 + y_{m+1}^2 - d_{m+1}^2)} \end{bmatrix} \tag{2}$$

$$X = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3) \\ \vdots & \vdots \\ 2(x_1 - x_m) & 2(y_1 - y_m) \\ \boxed{2(x_1 - x_{m+1})} & \boxed{2(y_1 - y_{m+1})} \end{bmatrix} \tag{3}$$

$$X^T X = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \tag{4}$$

$$a_{11} = 4\sum_{i=2}^{m} (x_1 - x_i)^2 + \boxed{4(x_1 - x_{m+1})^2} \tag{5}$$

$$a_{12} = a_{21} = 4\sum_{i=2}^{m} (x_1 - x_i)(y_1 - y_i) + \boxed{4(x_1 - x_{m+1})(y_1 - y_{m+1})} \tag{6}$$

$$a_{22} = 4\sum_{i=2}^{m} (y_1 - y_i)^2 + \boxed{4(y_1 - y_{m+1})^2} \tag{7}$$

$$X^T Y = \begin{bmatrix} 2\sum_{i=2}^{m} (x_1 - x_i) y\,[i-1] + \boxed{2(x_1 - x_{m+1}) y\,[m]} \\ 2\sum_{i=2}^{m} (y_1 - y_i) y\,[i-1] + \boxed{2(y_1 - y_{m+1}) y\,[m]} \end{bmatrix} \tag{8}$$

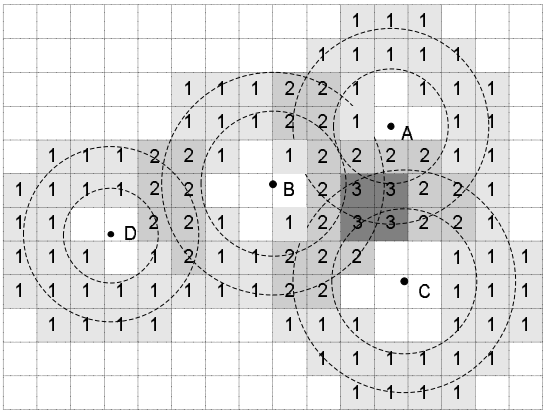


Fig. 5. The voting-based location estimation

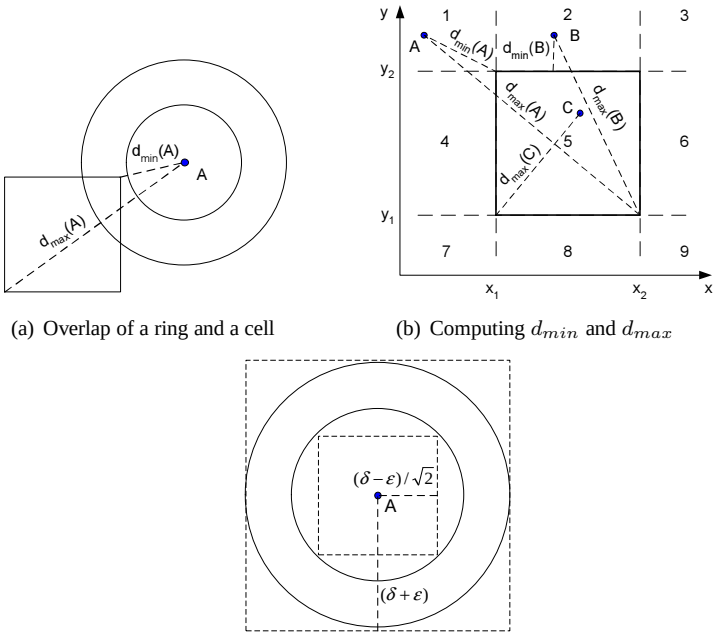
By reusing intermediate results, we can avoid a large number of redundant calculations, thus speeding up schemes such as EARMMSE.

3.4.3 *EARMMSE with Incremental Evaluation.* In EARMMSE, the location references are ordered in terms of the number of other "intersecting" location references. EARMMSE starts with the entire set of location references, and gradually removes the most suspicious location references based on the above order until an acceptable location estimation is derived.

To mostly efficiently use incremental ARMMSE, we perform EARMMSE in two rounds. In the first round, we start with three location references with the highest degree of consistency to perform location estimation. We then add a new location reference with the next highest degree of consistency, and complete the estimation using incremental ARMMSE method. We continue this process until all location references are used. We save location estimate and the corresponding minimum MSE for every step. In the second round, we check the minimum MSE for the sets of location references in the reverse order, similar to the original EARMMSE method. However, no MMSE computation is needed due to the saved results. The algorithm stops once we find a location estimate with a small enough minimum MSE. This implementation allows us to use incremental ARMMSE to perform EARMMSE efficiently without unnecessary, redundant computation.

4. VOTING-BASED LOCATION ESTIMATION

In this approach, we have each location reference "vote" on the locations at which the node of concern may reside. To facilitate the voting process, we quantize the target field into a grid of cells, and have each sensor node determine how likely it is in each cell based on each location reference. We then select the cell(s) with the highest vote and use the "center" of the cell(s) as the estimated location. To deal with the resource constraints on sensor nodes, we further develop an iterative refinement scheme to reduce the storage overhead, improve the accuracy of estimation, and make the voting scheme efficient on resource constrained sensor nodes.



(a) Overlap of a ring and a cell

(b) Computing d_{min} and d_{max}

(c) Limiting the examinations of cells

Fig. 6. Determine whether a ring overlaps with a cell

4.1 The Basic Scheme

After collecting a set of location references, a sensor node should determine the target field. The node does so by first identifying the minimum rectangle that covers all the locations declared in the location references, and then extending this rectangle by R_b, where R_b is the maximum transmission range of a beacon signal. This extended rectangle forms the target field, which contains all possible locations for the sensor node. The sensor node then divides this rectangle into M small squares (cells) with the same side length L, as illustrated in Figure 5. (The node may further extend the target field to have square cells.) The node then keeps a voting state variable for each cell, initially set to 0.

At the beginning of this algorithm, the non-beacon node needs to identify the candidate ring of each location reference. For example, in Figure 5, the ring centered at point A is a candidate ring at A, which is derived from the location reference with the declared location at A.

For each location reference $\langle x, y, \delta \rangle$, the sensor node identifies the cells that overlap with the corresponding candidate ring, and increments the voting variables for these cells by 1. After the node processes all the location references, it chooses the cell(s) with the highest vote, and uses its (their) geometric centroid as the estimated location of the sensor node.

4.2 Overlap of Candidate Rings and Cells

A critical problem in the voting-based approach is to determine if a candidate ring overlaps with a cell. We discuss how to determine this efficiently below.

Suppose we need to check if the candidate ring at A overlaps with the cell shown in Figure 6(a). Let $d_{min}(A)$ and $d_{max}(A)$ denote the minimum and maximum distances from a point in the cell to point A, respectively. We can see that the candidate ring does not overlap with the cell only when $d_{min}(A) > r_o$ or $d_{max}(A) < r_i$, where $r_i = \max\{0, \delta - \epsilon\}$ and $r_o = \delta + \epsilon$ are the inner and the outer radius of the candidate ring, respectively.

To compute d_{min} and d_{max}, we divide the target field into 9 regions based on the cell, as shown in Figure 6(b). It is easy to see that given the center of any candidate ring, we can determine the region in which it falls with at most 6 comparisons between the coordinates of the center and those of the corners of the cell. When the center of a candidate ring is in region 1 (e.g., point A in Figure 6(b)), it can be shown that the closest point in the cell to A is the upper left corner, and the farthest point in the cell from A is the lower right corner. Thus, $d_{min}(A)$ and $d_{max}(A)$ can be calculated accordingly. These two distances can be computed similarly when the center of a candidate ring falls into regions 3, 7, and 9.

Consider point B in region 2. Assume the coordinate of point B is (x_B, y_B). We can see that $d_{min}(B) = y_B - y_2$. Computing $d_{max}(B)$ is a little more complex. We first need to check if $x_B - x_1 > x_2 - x_B$. If yes, the farthest point in the cell from B must be the lower left corner of the cell. Otherwise, the farthest point in the cell from B should be the lower right corner of the cell. Thus, we have

$$d_{max}(B) = \sqrt{(\max\{x_B - x_1, x_2 - x_B\})^2 + (y_B - y_1)^2}.$$

These two distances can be computed similarly when the center of a candidate ring falls into regions 4, 6, and 8.

Consider a point C in region 5. Obviously, $d_{min}(C) = 0$ since point C itself is in the cell. Assume the coordinate of point C is (x_c, y_c). The farthest point in the cell from C must be one of its corners. Similarly to the above case for point B, we may check which point is farther away from C by checking $x_c - x_1 > x_2 - x_c$ and $y_c - y_1 > y_2 - y_c$. As a result, we get

$$d_{max}(C) = \sqrt{(\max\{x_c - x_1, x_2 - x_c\})^2 + (\max\{t_c - y_1, y_2 - y_c\})^2}.$$

Based on the above discussion, we can determine if a cell and a candidate ring overlap with at most 10 comparisons and a few arithmetic operations. To prove the correctness of the above approach only involves elementary geometry, and thus is omitted.

For a given candidate ring, a sensor node does not have to check all the cells for which it maintains voting states. As shown in Figure 6(c), with simple computation, the node can get the outer bounding box centered at A with side length $2(\delta + \epsilon)$. The node only needs to consider the cells that intersect with or fall inside this box. Moreover, the node can get the inside bounding box with simple computation, which is centered at A with side length $\sqrt{2}(\delta - \epsilon)$, and all the cells that fall into this box need not be checked.

4.3 Iterative Refinement

The number of cells M (or equivalently, the quantization step L) is a critical parameter for the voting-based algorithm. It has several implications to the performance of our approach. First, the larger M is, the more state variables a sensor node has to keep, and thus the more storage is required. Second, the value of M (or L) determines the precision of location estimation. The larger M is, the smaller each cell will be. As a result, a sensor node can determine its location more precisely based on the overlap of the cells and the candidate rings.

However, due to the resource constraints on sensor nodes, the granularity of the partition is usually limited by the memory available for the voting state variables on the nodes. This puts a hard limit on the accuracy of location estimation. To address this problem, we propose an *iterative refinement* of the above basic algorithm to achieve fine accuracy with reduced storage overhead.

In this version, the number of cells M is chosen according to the memory constraint in a sensor node. After the first round of the algorithm, the node may find one or more cells having the largest vote. To improve the accuracy of location estimation, the sensor node then identifies the smallest rectangle that contains all the cells having the largest vote, and performs the voting process again. For example, in Figure 5, the same algorithm will be performed in a rectangle which exactly includes the 4 cells having 3 votes. Note that in a later iteration of the basic voting-based algorithm, a location reference does not have to be used if it does not contribute to any of the cells with the highest vote in the current iteration.

Due to a smaller rectangle to quantize in a later iteration, the size of cells can be reduced, resulting in a higher precision. Moreover, a malicious location reference will most likely be discarded, since its candidate ring usually does not overlap with those derived from benign location references. For example, in Figure 5, the candidate ring centered at point D will not be used in the second iteration.

The iterative refinement process should terminate when a desired precision is reached or the estimation cannot be refined. The former condition can be tested by checking if the side length L of each cell is less than a predefined threshold S, while the latter condition can be determined by checking whether L remains the same in two consecutive iterations. The algorithm then stops and outputs the estimated location obtained in the last iteration. It is easy to see that the algorithm will fall into either of these two cases, and thus will alway terminate. In practice, we may set the desired precision to 0 in order to get the best precision.

5. SECURITY ANALYSIS

Both proposed techniques can usually remove the effect of the malicious location references from the final location estimation when there are more benign location references than the malicious ones. Theorem 1 shows that when the majority of location references are benign, the location estimation error of the attack-resistant MMSE is bounded if we can successfully identify the largest consistent set. Hence, to defeat the attack-resistant MMSE approach, the attacker has to distribute to a victim node more malicious location references than the benign ones, and control the declared locations and the physical features (e.g., signal strength) of beacon signals so that the malicious location references are considered consistent.

LEMMA 2. *Assume there are m benign location references and n malicious location references in a τ-consistent set. The location estimation error from this set of location references using MMSE is no more than $2R + \sqrt{\frac{m+n}{m}}\tau$, where R is the radio communication range of a sensor node.*

PROOF. Let $O = (x_0, y_0)$ denote the real location of the non-beacon node and $O' = (x'_0, y'_0)$ denote the estimated location of the non-beacon node based on all location references (including the malicious ones). Let $|AB|$ denote the distance between A and B. Thus, the location estimation error can be represented by $|OO'|$. Let $\{L_1, \cdots, L_m\}$ denote

the set of benign location references and $\{L_{m+1}, \cdots, L_{m+n}\}$ denote the set of malicious location references.

Consider a particular benign location references $L_i = \langle x_i, y_i, \delta_i \rangle$. Since the communication range of sensor nodes is R, we have $|OL_i| \leq R$. In addition, $e_i = \delta_i - |O'L_i|$ and $\delta_i \leq R$. Thus, we have

$$|OO'| \leq |OL_i| + |L_iO'| \leq R + \delta_i - e_i \leq 2R - e_i.$$

There are two different cases: $e_i \geq 0$ or $e_i < 0$. When $e_i \geq 0$, we have $|OO'| \leq 2R$. When $e_i < 0$, we have $|OO'| - 2R \leq -e_i$. Assume $|OO'| \geq 2R$, we have $e_i^2 \geq (|OO'| - 2R)^2$. Since $\{L_1, \cdots, L_{m+n}\}$ is τ-consistent, we have $\sum_{i=1}^{m+n} e_i^2 \leq (m+n)\tau^2$. Therefore,

$$m(|OO'| - 2R)^2 \leq \sum_{i=1}^{m} e_i^2 \leq \sum_{i=1}^{m+n} e_i^2 \leq (m+n)\tau^2.$$

Hence, we have $(|OO'| - 2R)^2 \leq \frac{(m+n)\tau^2}{m}$. It implies

$$|OO'| \leq 2R + \sqrt{\frac{m+n}{m}}\tau.$$

According to the above analysis, we can conclude that the statement in Lemma 2 is true. □

THEOREM 1. *Assume a non-beacon node receives m benign location references and n malicious location references, where $m > n$. The location estimation error at this non-beacon node using the attack-resistant MMSE scheme with the brute-force algorithm is no more than $2R + \sqrt{\frac{m}{m-n}}\tau$ if the threshold τ is set greater than the maximum distance error ϵ, where R is the radio communication range of a sensor node.*

PROOF. It is easy to know that the set of m benign location references is always τ-consistent if $\tau \geq \epsilon$. Thus, there are at least m location references in the largest consistent set. Assume there are k location references in the largest consistent set, where $k \geq m$. According to Lemma 2, we have

$$|OO'| \leq 2R + \sqrt{\frac{k}{k-n}}\tau \leq 2R + \sqrt{\frac{m}{m-n}}\tau.$$

□

Similarly, theorem 2 shows that when the majority of location references are benign, the location estimation error of the voting-based scheme is bounded. Hence, to defeat the voting-based approach, the attacker needs similar efforts so that the cell containing the attacker's choice gets more votes than those containing the sensor's real location.

THEOREM 2. *When the majority of location references at a non-beacon node are benign, the location estimation error at this non-beacon node using the voting-based scheme is no more than $2R + \sqrt{2}L$, where L is the side length of the cell.*

PROOF. Assume the real location of the sensor node is $O = (x_0, y_0)$ and the estimated location of the sensor node using the voting-based scheme is $O' = (x_0', y_0')$.

Since the candidate ring of a benign location reference always covers the real location O of the sensor node, the number of votes in the cell that contains O is at least m. Thus, the

number of votes in the cell that contains O' is at least m. Since the number of votes coming from the malicious location references is at most n, we know that there is at least one benign location reference whose candidate ring covers the cell that contains O'. Assume one of such benign location references is $L_i = \langle x_i, y_i, \delta_i \rangle$, we have

$$|L_i O'| \leq R + \sqrt{2}L,$$

where L is the side length of a cell. Therefore, we have

$$|OO'| \leq |OL_i| + |L_i O'| \leq 2R + \sqrt{2}L.$$

□

An attacker has two ways to satisfy the above conditions (in order to defeat our techniques). First, the attacker may compromise beacon nodes and then generate malicious beacon signals. Since all beacon packets are authenticated, and a sensor node uses at most one location reference derived from the beacon signals sent by each beacon node, the attacker needs to compromise more beacon nodes than the benign beacon nodes from which a target sensor node may receive beacon signals, besides carefully crafting the forged beacon signals.

Second, the attacker may launch wormhole attacks [Hu et al. 2003] (or replay attacks) to tunnel benign beacon signals from one area to another. In this case, the attacker does not have to compromise any beacon node, though he/she has to coordinate the wormhole attacks. This paper does not provide techniques to address wormhole attacks. However, our methods can still tolerate wormhole attacks to a certain degree as long as the number of malicious location references at a sensor node is less than the number of benign location references. Moreover, in another related work [Liu et al. 2005b], we have developed an approach for effectively detecting malicious beacon signals transmitted through wormholes on the current sensor platforms (with low quality clocks). Thus, the proposed techniques in this paper and the complementary detection techniques in [Liu et al. 2005b] can be integrated to effectively defend attacks against range-based localization in wireless sensor networks.

As discussed earlier, an attacker may also jam the communication channel to, for example, prevent the successful transmission and receiving of beacon signals. This is a common threat to all wireless networks. Thus, the approaches in this paper require additional mechanisms to ensure that the attacker cannot jam the communication channel continuously without being detected and removed.

Our techniques certainly have a limit. In an extreme case, if all the beacon nodes are compromised, our techniques will fail. However, the proposed techniques offer a graceful performance degradation as more malicious location references are introduced. In contrast, an attacker may introduce arbitrary location error with a single malicious location reference in the previous schemes. To further improve the security of location discovery, other complementary mechanisms (e.g., detection of malicious beacon nodes [Liu et al. 2005b], anti-jamming techniques) should be used.

6. EXPERIMENTAL EVALUATION

We have given a preliminary experimental evaluation in the preliminary version of this paper [Liu et al. 2005a]. In this section, we present a thorough experimental evaluation to validate the new techniques proposed in this paper and compare existing secure and

resilient localization schemes that can be used on the current generation of sensor platforms (e.g., MICAz and MICA2 motes). Our evaluation uses both outdoor experiments in a testbed of MICAz motes and simulation evaluation using channel profiles obtained from the testbed. The schemes under evaluation include the ARMMSE schemes, voting based scheme, and Least Median of Squares (LMS) based location estimation [Li et al. 2005].

There are several other secure and resilient localization schemes, including SeRLoc [Lazos and Poovendran 2005], SPINE [Capkun and Hubaux 2005], and ROPE [Lazos et al. 2005]. However, SeRLoc requires directional antenna on sensor nodes, SPINE requires nano-second scale time synchronization among nodes, and ROPE, which is an integration of SeRLoc and SPINE, requires both directional antenna and nano-second scale time synchronization. These requirements cannot be met on the current generation of sensor platforms such as MICA series of motes. Thus, we cannot include them in this evaluation.

6.1 Evaluation Methodology

6.1.1 *Evaluation Criteria.* Key issues with localization service in wireless sensor networks are resiliency and efficiency. For an unattended wireless sensor network, the best defense against attacks is to build strong resiliency against attacks, even when nodes are compromised. Given that a sensor node is severely constrained in both hardware resources and battery power, it is extremely important to develop efficient algorithms. Under these key requirements, we use four criteria to evaluate the secure and resilient localization schemes under study.

First, we evaluate the algorithm implementation in terms of ROM and RAM usage. ROM is used to store the compiled code. The ROM size reflects the total non-volatile storage that is needed to store the localization code. RAM usage reflects the runtime memory requirement. Since both ROM and RAM are premier resources on a resource constrained sensor node (for example a MICAz mote has an Atmel ATmega 128 processor with 4KB RAM and 128KB ROM), it is highly desirable for the localization algorithm to have as small a footprint as possible.

We next compare the execution time of the localization algorithms. Localization schemes need to be implemented efficiently. A fast execution time translates to a low energy consumption, which extends the lifetime of a sensor node. In addition, a sensor network usually needs to carry out specific applications. It is always desirable to have network services such as localization consume as few CPU cycles as possible.

The third, and the most important criteria is the resiliency of an algorithm against different levels of attacks. In this study, we assume that message exchanges are cryptographically protected, and false injection of localization information is eliminated. Moreover, we assume there are mechanisms that detect replayed localization messages, such as the one proposed in [Liu et al. 2005b]. Thus, the focus is on the robustness and resiliency against malicious location references. Obviously the more malicious location references, the harder it is for the resilient algorithm to defend. We evaluate the level of resiliency of each localization algorithm in terms of localization accuracy. We confirm (the theoretical analysis in the previous section) experimentally that our schemes may tolerate close to half of malicious location references.

In addition, to correlate the study on localization accuracy, we also investigate the success rate of each scheme to detect and filter out malicious location references. When the success rate is 100%, the location estimation error is solely due to the measurement error caused by the radio channel. When the success rate is low, some malicious location refer-

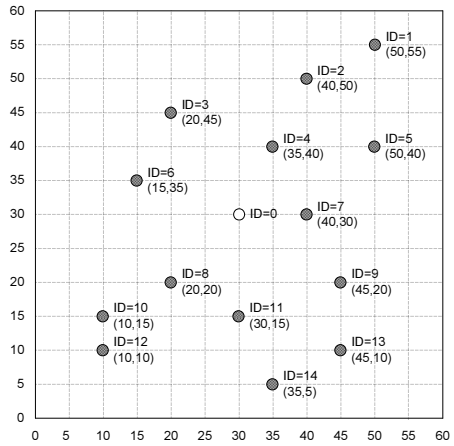


Fig. 7. Target field (60feet $\times$ 60feet)

ences are not filtered out. Their injected errors could significantly distort the final location estimation.

6.1.2 *Field Experiments using MICAz Motes.* We perform a series of outdoor field experiments using MICAz motes to compare the secure and resilient localization schemes discussed earlier. These field experiments offer an opportunity to observe their performance in a realistic setting.

We implemented the ARMMSE and the voting-based schemes in nesC on TinyOS. We also implemented the LMS method based on matlab code obtained from the authors of [Li et al. 2005]. We use the RSSI method to measure the distance, since this is the only option for MICAz motes.

In the outdoor field experiments, we deploy 15 MICAz motes in a 60feet $\times$ 60feet target field, as shown in figure 7. We use up to 14 motes as beacon nodes to replicate a dense deployment. The beacon nodes broadcast location reference messages periodically. The sensor node with ID 0 (in the middle of the field) is a representative of a non-beacon node that needs its own location.

With this deployment setup, we perform experiments under four attack scenarios. In the first scenario, one randomly selected beacon node is configured as being malicious, which reports a faulty location reference x feet away from its true location in a random direction. In the second scenario, we randomly pick up four malicious beacon nodes. Each malicious beacon node adds a random location offset of x feet from its true location. The third scenario mimics node collusion. Four randomly selected beacon nodes collude with each other and send out false but consistent location references. In this case, all malicious beacon nodes report a falsified position shifted x feet from its true location in the same direction. In the fourth scenario, we experiment with a varying number of colluding nodes ranging from 1 to 8 (out of 14 beacon nodes) to examine the impact on the estimated location.

Under each attack scenario, we investigate the algorithm resiliency in terms of localization error and malicious location reference detection rate, and the algorithm efficiency in terms of execution time. In the experiment we vary the error injected from 10 to 150 feet

with a 10 feet increment.

6.1.3 *TOSSIM Simulation.* Due to the cost of conducting field experiments, we cannot perform the evaluation with many different deployments using different nodes. To understand the performance in a wider variety of situations, we resort to simulation through TinyOS simulator TOSSIM [Levis et al. 2003]. TOSSIM is a discrete event simulator that can be used to run TinyOS applications, aimed at providing a fidelity simulation for TinyOS applications by simulating system interrupts and the network at the bit level.

Profiling of MICAz Radio Channel: Carrying out a simulation faithfully to a real deployment experiment requires that the simulator incorporate sensor characteristics such as timing model and radio channel model. Currently, TOSSIM does not provide a completely true representation of a real network. For example, it does not model the node variations typically seen in low cost sensor nodes, nor does it incorporate real world signal distortion through reflection or attenuation. Since we use the RSSI method to obtain distance measurement in the current performance evaluation study, we do need to incorporate proper radio channel characteristics in the simulation. Otherwise, the simulation results could significantly deviate from those obtained through field experiments.

We use a set of 30 MICAz motes to profile the channel characteristics for the purpose of RSSI-based distance measurement. Due to node and environmental variations, one radio channel between two nodes may differ from each other. Profiling per-radio channel characteristics between each pair of sender and receiver is useful when there is a need to correct/compensate per-channel errors. However, getting the channel profile between any two nodes (a total of 30*29/2 possible pairs) would be cost prohibitive. So instead we seek to obtain the average channel profile for the nodes. To obtain average channel profile, we randomly draw 7 nodes from the pool of 30 node. We select one node as the receiver, and use the rest 6 nodes as the transmitter and place them at 10, 20, 30, 40, 50, and 60 feet away, respectively, from the receiver. From the transmitter, 1,000 reference messages are sent to each receiver. We use the received signal strength to "measure" the *known* node distance and obtain a total of 6 sets of 1,000 distance measurements at 10, 20, ..., 60 feet distances, respectively. We then rotate the positions of the 6 senders, so that we have the data for each node at each distance. We repeat the same experiment 5 times, using a different batch of 7 nodes randomly drawn. To include the impact of battery variations, we use different batteries with voltage between 2.7V and 3.1V. These experiments give 30,000 distance measurement samples with different battery voltages and different pairs of nodes at each of the 6 distances, providing us a fairly good average channel profile. These results are then incorporated into our simulation to mimic the channel characteristics.

We build a distance measurement error model based on the above data. For each of the above distances, we use the median RSSI as the reference to convert an RSSI reading to the given distance, and use interpolation for the other distances. Based on this conversion and the collected RSSI readings, we then build error distributions for 10, 20, ..., and 60 feet. Figure 8 shows the distance measurement error model for these distances. The error distributions for the other distances are calculated using interpolation based on these distributions. In our simulation, distance measurement errors are generated based on the error distributions in this model.

Simulation Setup: As discussed earlier, the simulation is aimed at evaluating the performance of these secure and resilient localization schemes with different combinations of deployments and distance measurement errors. In our simulations, we adopt the same

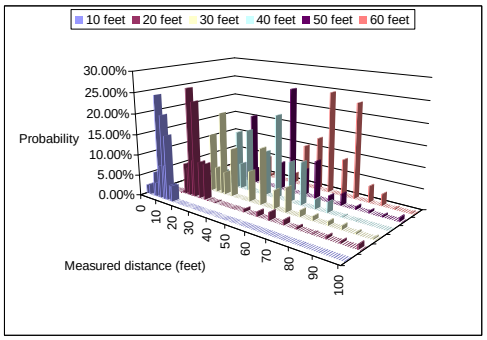


Fig. 8. Measured distance distribution using interpolation method

target field setting used in the field experiment. We run two sets of simulations. The first set of simulation replicates the same layout of beacon nodes as in the field test. In the second set of simulations, a set of 14 beacon nodes are randomly deployed in the target field. The non-beacon node, which needs to to be localized, is placed at the center of the target field. The beacon nodes broadcast localization messages periodically. The non-beacon node estimates its distance from each beacon node. The distance measurement error is simulated with the channel profile we obtained through experiments. In each of the scenarios discussed in Section 6.1.1, we run 1,000 rounds of simulation in TOSSIM to obtain the performance measures.

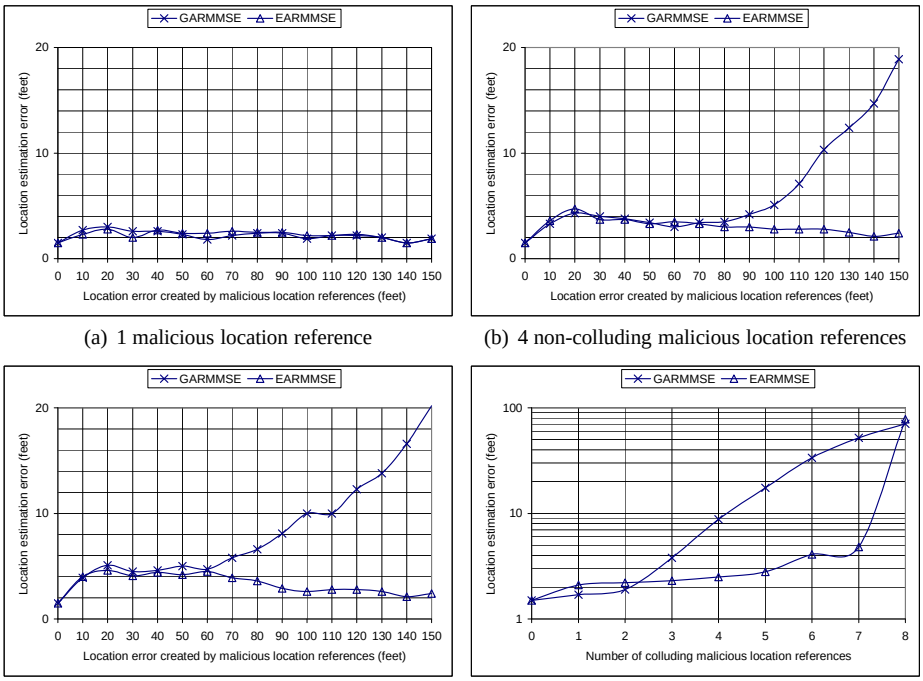
Given a maximum number of location references, all the schemes under investigation require a few parameters to be configured. These parameters must be set appropriately to ensure a fair comparison. For the EARMMSE scheme, we set the mean square error threshold $\tau = 0.8\epsilon$ as discussed earlier, where ϵ is the maximum distance measurement error. Based on the channel profiling results, $\epsilon = 7.4feet$. Thus, we set $\tau = 0.8\epsilon = 5.92feet$.

The critical parameter for the voting-based scheme is the number of cells M in the grid in each iteration. The cell number needs be the square of an integer in our experiments. The voting-based scheme is in general much slower than the EARMMSE scheme, no matter how we configure M. To ensure a fair comparison, we set the parameter M in the voting-based scheme in such a way that they consume a similar amount of RAM. As shown in table I, we set $M = 15^2 = 225$ to match the RAM consumption in the EARMMSE scheme. For further comparison, we also obtain the result with $M = 100$ besides $M = 225$ for the voting-based scheme.

Similar to the voting-based scheme, the LMS scheme [Li et al. 2005] is much slower than the EARMMSE scheme. To ensure fair comparison, we set the subset size $n = 4$. Moreover, we set the number M of subsets to be examined in such a way that the LMS scheme has the same average execution time as the voting-based scheme when the grid size is 100 and 225, respectively. As shown in table I, LMS has a fixed RAM consumption, which is similar to the RAM consumption in the voting-based scheme.

6.2 Effectiveness of EARMMSE and Incremental Evaluation

In this subsection, we focus on the experimental evaluation of EARMMSE and the incremental evaluation of EARMMSE. As we will see in the evaluation results, these techniques



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

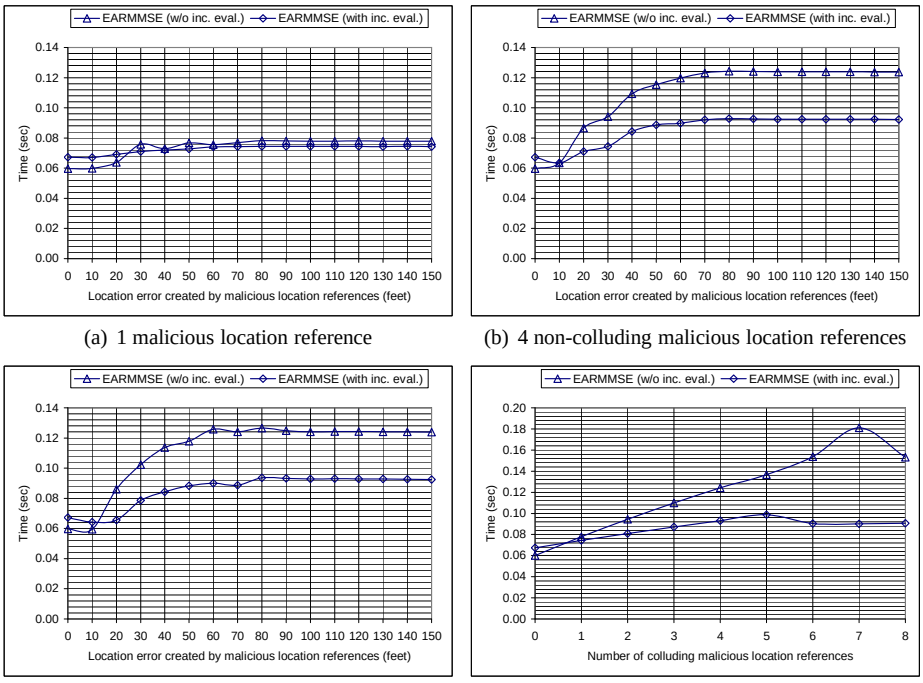
(d) varying number of colluding malicious location references (error created by malicious location reference is 100 feet)

Fig. 9. Location estimation error for GARMMSE and EARMMSE

significantly outperform the previous techniques proposed in the preliminary version of this paper [Liu et al. 2005a].

Figure 9 shows the location estimation errors of both GARMMSE and EARMMSE in the four evaluation scenarios. As we can see in figure 9(a), both approaches can effectively defeat 1 malicious location reference. However, as shown in figures 9(b) and 9(c), GARMMSE cannot effectively bound the location error introduced by 4 colluding or non-colluding malicious location references. (This result is consistent with the preliminary evaluation reported in [Liu et al. 2005a].) In contrast, EARMMSE can effectively defeat both 4 non-colluding and 4 colluding malicious location references. Figure 9(d) further shows EARMMSE performs significantly better than GARMMSE in tolerating colluding malicious location references. All the above results demonstrate that EARMMSE is much more effective than GARMMSE in bounding location estimation errors.

We also perform experiments to confirm the effectiveness of incremental evaluation in reducing the computational cost of EARMMSE. We use EARMMSE both with and without incremental evaluation in the four evaluation scenarios. Figure 10 shows the timing results. When there is only 1 malicious location reference, incremental evaluation slightly reduces the execution time, as shown in figure 10(a). When there are 4 colluding or non-colluding malicious location references, incremental evaluation can reduce the execution time by about 50%, as shown in figures 10(b) and 10(c). Figure 10(d) further shows that incremental evaluation reduces more as there are more colluding malicious location ref-



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding location references

(d) varying number of colluding malicious location references (error created by malicious location reference is 100 feet)

Fig. 10. Effectiveness of incremental evaluation

erences. Indeed, the more malicious location references, the more execution time incremental evaluation can save. When the number of colluding malicious location references reaches 8, the execution time for EARMMSE starts to drop. This is the point where EARMMSE fails; there are more colluding malicious location references than the benign ones, and the benign ones are considered "malicious" and removed.

The code sizes of GARMMSE and EARMMSE (with and without incremental evaluation) are shown in table I. EARMMSE does have more ROM size than GARMMSE (442 bytes), and incremental evaluation further increases both ROM and RAM sizes (282 bytes and 148 bytes, respectively). However, the increases in ROM and RAM sizes are still affordable on current sensor platforms such as MICAz motes.

Due to the effectiveness of EARMMSE and incremental evaluation, we will use EARMMSE with incremental evaluation as the representative of ARMMSE schemes in our later comparison. By default, we assume an EARMMSE implementation includes incremental evaluation, unless specified otherwise.

6.3 Comparison of Alternative Approaches

We now compare the various approaches for attack-resistant location estimation according to the evaluation methodology discussed in Section 6.1, including basic MMSE (as a reference), EARMMSE (with incremental evaluation), voting-based scheme, and the LMS based approach [Li et al. 2005]. Our goal is to identify the scheme most suitable for the

current generation of sensor platforms.

6.3.1 *Code Size Comparison.* Table I gives the code size of each localization algorithm implementation. The code was compiled for MICAz, assuming that each scheme handles up to 14 location references. These numbers were obtained using the *check_size.pl* script provided in the TinyOS distribution. Note that the RAM size does not include the memory consumed by local variables on the stack. Since nesC Compiler optimizes the whole program by default, to compare the size of code fairly, we disabled any optimization of nesC compiler. From the table it is obvious that the voting-based scheme uses the most ROM, while all schemes use a similar amount of RAM except for the voting-based scheme when $M = 100$.

Table I. Code Size (up to 14 location references)

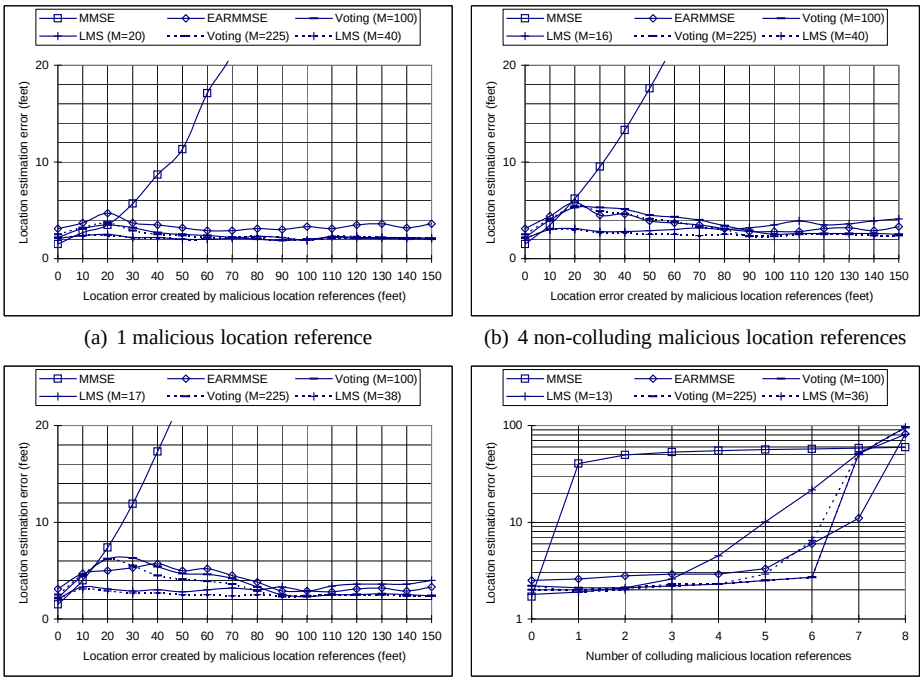
Scheme	ROM (bytes)	RAM (bytes)
Basic MMSE	2,734	0
GARMMSE	4,422	100
EARMMSE (w/o inc. eval.)	4,864	100
EARMMSE	5,146	248
Voting (M=100)	7,074	100
Voting (M=225)	7,074	225
LMS	5,262	237

6.3.2 *Results of Field Experiments.* Figures 11, 12, and 13 show the location estimation error, the execution time, and the success rate of removing malicious location references for all schemes under study, respectively.

From figures 11(a), 11(b), and 11(c), we can see that all resilient schemes under study have bounded location estimation error under each attack scenario while the basic MMSE scheme [Savvides et al. 2001] cannot tolerate even one malicious location reference. For the basic MMSE method, the location estimation error increases with the increase of the injected error. Although the three schemes use completely different approaches to defend against malicious location references, the level of tolerance (in terms of localization error) of each scheme to the injected error is fairly comparable under the first three attack scenarios. In general (from Figures 11(a), 11(b), and 11(c)) the LMS scheme provides a slightly lower location estimation error in comparison to the voting-based and the EARMMSE schemes, but the performance results are fairly close.

Another interesting discovery is that the largest location estimation error happens when the injected error is small (around 20 feet)! This phenomena can be observed almost on every scheme and under each attack scenario. Although counter intuitive, this observation can be explained. Attack-resistant localization schemes typically rely on outlier detection or consistency check to remove malicious location references before performing the final location calculation. When the injected error is small, especially on the same scale or close to the range of measurement error, these schemes are not able to effectively distinguish and remove the malicious ones. This leads to the enlarged location estimation error.

To support our explanation, we investigate the effectiveness of each scheme to filter out malicious location references under different amounts of error injection. For each scheme under every scenario, we capture the number of malicious location references that have been successfully identified in each round and calculate the average detection rate



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

(d) varying number of colluding malicious location references (error created by malicious location reference is 100 feet)

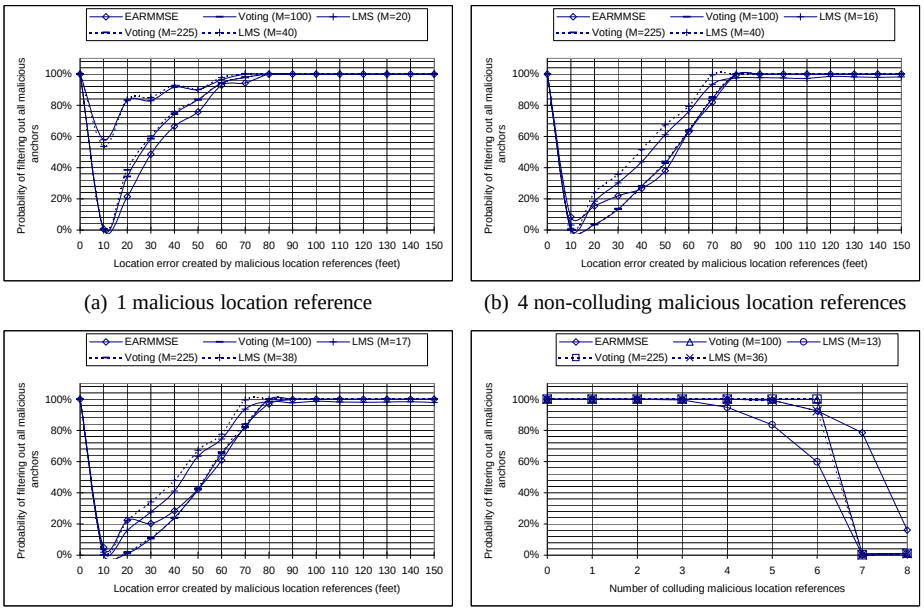
Fig. 11. Location estimation error in field experiments

over 1,000 rounds. Figures 12(a), 12(b), and 12(c) show the probability of successfully removing malicious location references in our experiments.

All three schemes fail to identify and remove malicious location references when the injected errors are small ($<$70 feet). When the injected error is at 10 feet, no scheme is able to identify and remove the malicious location references. Nevertheless, the injected errors in such cases are very close to normal measurement errors, and do not introduce significant errors into location estimation.

Figure 12(d) provides the results on the malicious location reference detection probability when we have 1$\sim$8 malicious location references. Since the injected error is 100 feet, all schemes should be able to identify and remove the malicious location references when the number of colluding ones is small (1$\sim$6). The Figure shows that the LMS scheme is the first scheme to break down while the EARMMSE method provides the best detection rate which translates to the best resiliency in our experiment.

We also evaluate the algorithm resiliency in terms of how many malicious location references a scheme can tolerate. Setting the injected error to 100 feet, we experiment algorithm resiliency by introducing 1$\sim$8 malicious location references out of a total of 14 ones. Figure 11(d) displays the result of this study. It shows that all resilient schemes perform well when we introduce up to 6 malicious location references. When half of them are malicious, the EARMMSE method can still maintain an estimation error of 10 feet, while the voting-based scheme and the LMS scheme fail. Obviously, when more than half of the location



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

(d) varying number of colluding malicious location references (error created by malicious location references is 100 feet)

Fig. 12. Success rate of removing malicious location references in field experiments

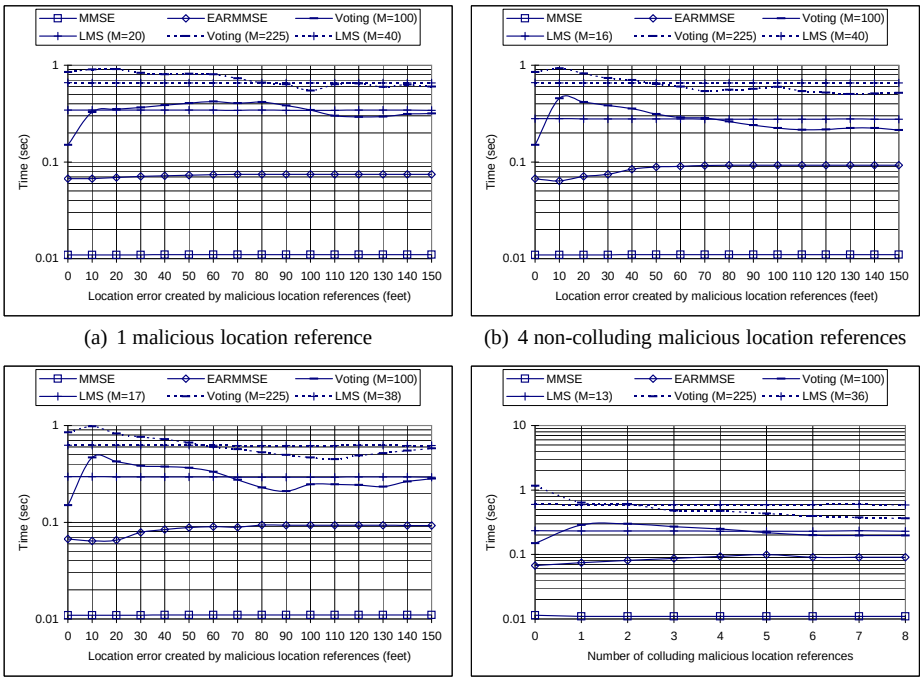
references are malicious, none of the scheme would work. The figure also shows that the LMS scheme is sensitive to the selection of number of subsets $\boldsymbol{M}$. When M is set to 13, the LMS scheme begin to fail with only 4 malicious location references.

We next perform study on efficiency in terms of execution time. Among all the schemes evaluated, both the LMS and the basic MMSE schemes have a fixed execution time. The execution time of the LMS scheme depends on the number of random subset (M) and size of subset (n). The EARMMSE and the voting-based schemes have a variable runtime.

From Figure 13, we can see that the EARMMSE scheme has a much smaller running time ($<$0.1 seconds) in comparison to the voting-based scheme or the LMS scheme (around 1 second). The basic MMSE scheme has the shortest running time, but it provides no resilience at all. Among all the resilient schemes evaluated, the EARMMSE scheme is about 10 times faster than the others.

From Figures 13(b) and 13(c), another observation is that the peak computation time is observed when low error is injected by the malicious location references. This is consistent to our earlier discussion on localization error, which peaks when the injected errors are small. The algorithms have a tough time trying to identify the malicious location references.

6.3.3 *Results of Simulation.* We carry out two sets of simulations. The first set of simulation duplicates the exact topology and attack scenarios as those in the field test, which was described in Section 6.1.2. To make a fair comparison, the same parameter setup for each scheme is used. We also incorporate the channel profile we obtained through



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

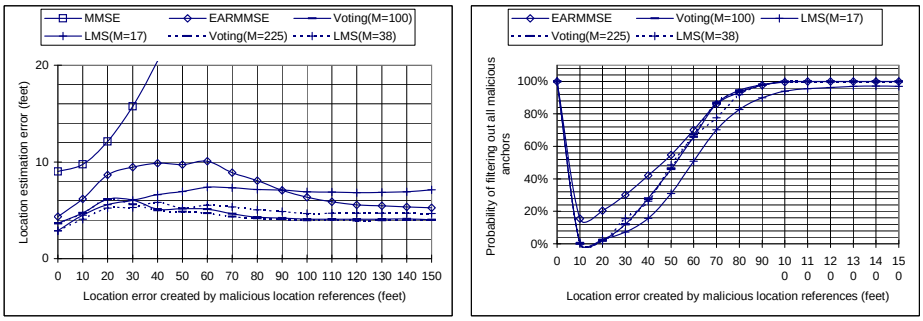
(d) varying number of colluding malicious location references (error created by malicious location references is 100 feet)

Fig. 13. Execution time in field experiments

experiments into simulation.

The results of the first set of simulation are in general consistent with those of the field experiments. For space reasons, we do not include the performance results for all scenarios. Figure 14(a) shows the location estimation error for the attack scenario with 4 colluding nodes. The location estimation errors are in general larger that those in the field experiments. This is because the randomly generated distance measurement errors are on average larger than those encountered in the field experiments. Overall, the trend matches the results in the field experiments very well. Except for the basic MMSE method, all resilient schemes have bounded location estimation error under each attack scenario. In addition, all three schemes under study have comparable performance in terms of location estimation error under non-colluding and colluding attacks.

The success rate of filtering malicious location references is another important criterion we use to evaluate resiliency. Figure 14(b) provides results on how successful of each scheme can filter out malicious location references in the 4 colluding node attack scenario. Similar to the field experiments, we can observe that none of the resilient schemes can deal with small error injection well. When the injected error is small, a malicious location reference may successfully blend itself among the rest. When a resilient localization scheme fails to filter out malicious location references, the injected error directly contributes to location estimation error. For example, in Figure 14(a) location estimation errors peak in the low range of the injected errors.



(a) 4 colluding malicious location references (b) 4 colluding malicious location references

Fig. 14. Simulation results using the field experiment deployment

Topology may affect localization error potentially. (As an extreme example, one cannot use three beacon nodes in a co-linear layout to perform localization.) In the second set of simulation, we investigate how each scheme performs without the influence of a particular topology. To achieve that, for each round of experiment, the 14 location references are randomly positioned. We run a total of 1,000 rounds of experiments. The location estimation error and the success rate of filtering out malicious location references are obtained using the average of the 1,000 rounds of experiments.

Figure 15 and figure 16 show the location estimation error and the rate for successful filtering, respectively.

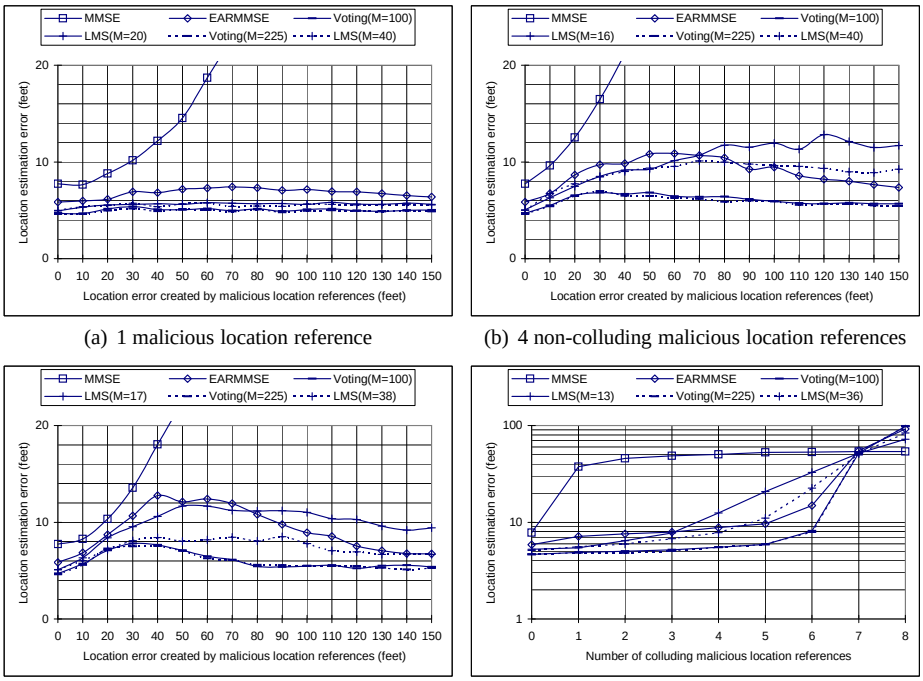
By comparing Figure 14(a) with Figure 15(c), and Figure 14(b) with Figure 16(c), we can observe that the results from the two sets of simulations are very similar to each other. For example, under both sets of simulation, the worst performance of malicious location reference detection and filtering happen when the injected error is at 10 feet. This observation confirms that the topology we use in field experiments is a typical deployment scenario and the topology of this deployment has not influenced much the localization estimation accuracy or the rate of successful filtering.

We make another observation from the two sets of simulations. Although fairly comparable, the EARMMSE scheme provides a slightly better result in terms of malicious location reference detection and elimination over the voting-based and the LMS schemes, as can be seen in both figure 14(b) and figure 16.

6.4 Discussion

In this experimental evaluation, we compared all existing range-based secure and resilient localization schemes suitable for the current generation of sensor nodes, through simulation and field experiments using MICAz motes as a test platform. These include the LMS scheme, the EARMMSE scheme, and the voting-based scheme. To facilitate the simulation study, we performed substantial experiments to profile the radio characteristics, which was incorporated in the simulation experiments.

Results from the field experiments generally matched well with the two sets of simulations we performed. This confirms that the channel profile we incorporated into the simulation has enabled us to capture the characteristics of the radio channels effectively. The first simulation confirms our field experiment results while the second set of simulation using a random deployment of beacon nodes gives us a high-confidence results, since



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

(d) varying number of colluding malicious location references (error created by malicious location references is 100 feet)

Fig. 15. Location estimation error in simulation (random placement of beacon nodes)

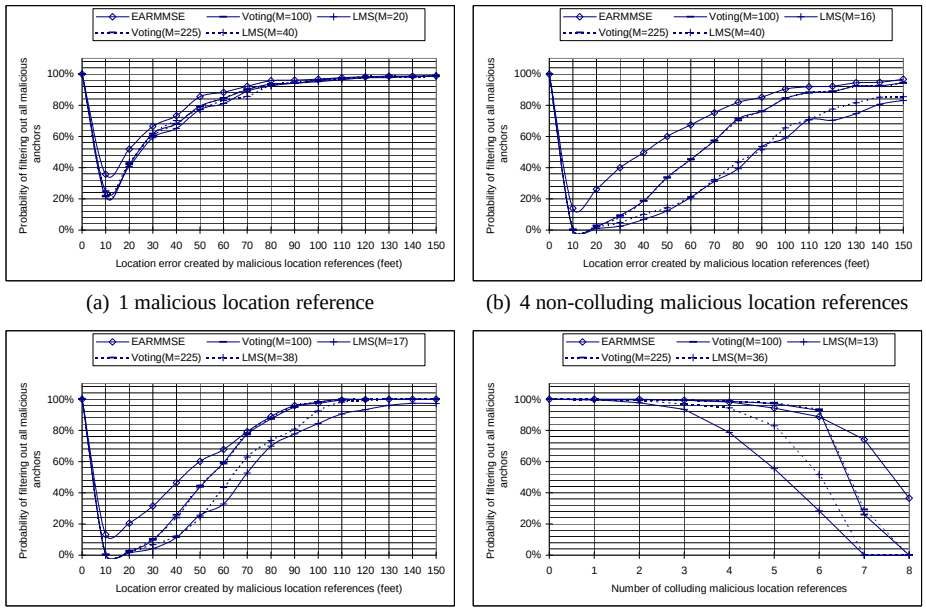
it eliminates the potential influence from beacon node placement.

Both field test and simulation lead to the following conclusions: 1) The current class of resilient algorithms that can be implemented on MICAz motes can provide reasonable resiliency against malicious location references; the location errors are well bounded, and these schemes can resist close to half of the location references being malicious. 2) Resilient localization schemes can deal with large injected errors effectively, but they have a hard time identifying malicious location references when the injected errors are small. Nevertheless, when the injected errors are small, they are very close to normal measurement errors, and do not introduce significant error into the final location estimation.

Based on our efficiency and resiliency criteria, we can conclude that the EARMMSE scheme has the shortest execution time while providing a similar resiliency to the other schemes. Thus, it is well suited for wireless sensor network applications running on current resource constrained sensor platforms such as MICAz motes.

7. RELATED WORK

Many range-based localization schemes have been proposed for sensor networks [Savvides et al. 2001; Savvides et al. 2002; Niculescu and Nath 2003a; Nasipuri and Li 2002; Doherty et al. 2001]. Savvides et al. developed AHLoS protocol based on Time Difference of Arrive [Savvides et al. 2001], which was extended in [Savvides et al. 2002]. Doherty et al. presented a localization scheme based on connectivity constraints and relative signal



(a) 1 malicious location reference

(b) 4 non-colluding malicious location references

(c) 4 colluding malicious location references

(d) varying number of colluding malicious location references (error created by malicious location references is 100 feet)

Fig. 16. Success rate of removing malicious location references in simulation (random placement of beacon nodes)

angles between neighbors [Doherty et al. 2001]. Angle of Arrival is used to develop localization scheme in [Niculescu and Nath 2003a] and [Nasipuri and Li 2002]. Range-free schemes are proposed to provide localization services for the applications with less precision requirements [Bulusu et al. 2000; Niculescu and Nath 2003b; Nagpal et al. 2003; He et al. 2003]. Bulusu, Heidemann and Estrin proposed to estimate a sensor's location as the centroid of all locations in the received beacon signals [Bulusu et al. 2000]. Niculescu and Nath proposed to use the minimum hop count and the average hop size to estimate the distance between nodes and then determine sensor nodes' locations accordingly [Niculescu and Nath 2003b]. None of these schemes will work properly when there are malicious attacks.

The location verification technique proposed in [Sastry et al. 2003] can verify the relative distance between a verifying node and a sensor node. It does not provide a solution to conduct secure location estimation at non-beacon nodes. In this paper, we provide efficient ways to estimate locations of sensor nodes securely. The location verification technique is complementary to our techniques since it can be used to enhance the security of distance measurement between two nodes.

A robust location detection is developed in [Ray et al. 2003]. However, it cannot be directly applied in sensor networks due to its high computation and storage overheads. A voting-based Cooperative Location Sensing (CLS) was proposed in [Fretzagias and Papadopouli 2004]. However, CLS is designed for powerful nodes (e.g., PDAs), while our scheme further uses iterative refinement to improve the performance with small storage

overhead. Therefore, our technique can be implemented and executed efficiently on resource constrained sensor nodes.

Similar to our attack-resistant location estimation techniques, the following two techniques are independently discovered to tolerate malicious attacks against location discovery in wireless sensor networks. A robust statistical methods that is similar to the attacker-resistant MMSE scheme is discovered in [Li et al. 2005] to achieve robustness through Least Median of Squares. A secure range-independent localization scheme (SeRLoc) that is similar to our voting-based scheme is discovered in [Lazos and Poovendran 2004] to protect location discovery with the help of sectored antennae at beacon nodes. Compared to these two studies, we provide more alternative ways to tolerate malicious attacks and also include the real implementation and field experiments in this paper.

SPINE [Capkun and Hubaux 2005] is developed to protect location discovery by using verifiable multilateration. However, the distance bounding techniques required for verifiable multilateration may not be available due to the difficulties to (1) deal with the external replay attacks in Ultrasound-based distance bounding and (2) achieve nanosecond processing and time measurements in Radio-based distance bounding. ROPE [Lazos et al. 2005] is developed by integrating SeRLoc and SPINE. However, it still requires nanosecond processing and time measurements that are not desirable for the current generation of sensor networks. Compared with these two studies, we provide techniques to tolerate malicious attacks without the above constraints. Moreover, our proposed techniques can be easily combined with most of existing localization techniques.

To further enhance the security of location discovery, a practical technique is developed to detect malicious beacon nodes that are providing malicious beacon signals [Du et al. 2005; Liu et al. 2005b]. This detection technique can be easily combined with our techniques. We consider it complementary to the techniques in this paper.

In addition to secure location discovery, location privacy becomes a more and more interesting topic recently. Several techniques are developed recently to protect the location privacy in sensor networks [Ozturk et al. 2004; Kamat et al. 2005].

Security in sensor networks has attracted a lot of attention in the past several years. To provide practical key management, researchers have developed key pre-distribution techniques [Eschenauer and Gligor 2002; Chan et al. 2003; Du et al. 2003]. To enable broadcast authentication, a protocol named μTESLA has been explored to adapt to resource constrained sensor networks [Perrig et al. 2001]. Security of sensor data has been studied in [Przydatek et al. 2003; Hu and Evans 2003]. Attacks against routing protocols in sensor networks and possible counter measures were investigated in [Karlof and Wagner 2003]. The research in this paper addresses another fundamental security problem that has not drawn enough attention.

8. CONCLUSION

In this paper, we developed several attack-resistant MMSE-based location estimation techniques and a voting-based location estimation technique to deal with attacks in localization schemes. The final schemes, the EARMMSE scheme with incremental evaluation and the voting-based scheme are both effective. We also performed experimental evaluation of all the secure and resilient location estimation schemes that can be used on the current generation of sensor platforms, through both simulation and field experiments with a network of MICAz motes. Our evaluation indicated that the EARMMSE scheme with incremen-

tal evaluation is most suitable for the current sensor platforms among all the alternative approaches.

Acknowledgment

The authors would like to thank Wade Trappe for providing the Matlab implementation of the LMS based location estimation scheme in [Li et al. 2005], which has been used to confirm our nesC implementation on TinyOS. The authors would also like to thank Adrian Perrig and the anonymous reviewers for their valuable comments.

REFERENCES

AKYILDIZ, I., SU, W., SANKARASUBRAMANIAM, Y., AND CAYIRCI, E. 2002. Wireless sensor networks: A survey. *Computer Networks 38*, 4, 393–422.

BULUSU, N., HEIDEMANN, J., AND ESTRIN, D. 2000. GPS-less low cost outdoor localization for very small devices. In *IEEE Personal Communications Magazine*. 28–34.

CAPKUN, S. AND HUBAUX, J. 2005. Secure positioning of wireless devices with application to sensor networks. In *Proceedings of IEEE InfoCom'05 (to appear)*.

CHAN, H., PERRIG, A., AND SONG, D. 2003. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Research in Security and Privacy*. 197–213.

CROSSBOW TECHNOLOGY INC. Wireless sensor networks. `http://www.xbow.com/Products/Wireless_Sensor_Networks.htm`.

DOHERTY, L., PISTER, K. S., AND GHAOUI, L. E. 2001. Convex optimization methods for sensor node position estimation. In *Proceedings of INFOCOM'01*.

DU, W., DENG, J., HAN, Y. S., AND VARSHNEY, P. 2003. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*. 42–51.

DU, W., FANG, L., AND NING, P. 2005. Lad: Localization anomaly detection for wireless sensor networks. In *Proceedings of the 19th IEEE International Parallel & Distributed Processing Symposium (IPDPS '05)*.

ESCHENAUER, L. AND GLIGOR, V. D. 2002. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*. 41–47.

FRETZAGIAS, C. AND PAPADOPOULI, M. 2004. Cooperative location-sensing for wireless networks. In *Second IEEE International conference on Pervasive Computing and Communications*.

GAY, D., LEVIS, P., VON BEHREN, R., WELSH, M., BREWER, E., AND CULLER, D. 2003. The nesC language: A holistic approach to networked embedded systems. In *Proceedings of Programming Language Design and Implementation (PLDI 2003)*.

HE, T., HUANG, C., BLUM, B. M., STANKOVIC, J. A., AND ABDELZAHER, T. F. 2003. Range-free localization schemes in large scale sensor networks. In *Proceedings of ACM MobiCom 2003*.

HILL, J., SZEWCZYK, R., WOO, A., HOLLAR, S., CULLER, D., AND PISTER, K. S. J. 2000. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*. 93–104.

HU, L. AND EVANS, D. 2003. Secure aggregation for wireless networks. In *Workshop on Security and Assurance in Ad Hoc Networks*.

HU, Y., PERRIG, A., AND JOHNSON, D. 2003. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of INFOCOM 2003*.

INTANAGONWIWAT, C., GOVINDAN, R., AND ESTRIN, D. 2000. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the sixth annual international conference on Mobile computing and networking (Mobicom '00)*. 56–67.

KAMAT, P., ZHANG, Y., TRAPPE, W., AND OZTURK, C. 2005. Enhancing source-location privacy in sensor network routing. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS '05)*. 599–608.

KARLOF, C. AND WAGNER, D. 2003. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of 1st IEEE International Workshop on Sensor Network Protocols and Applications*.

KARP, B. AND KUNG, H. T. 2000. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of ACM MobiCom 2000*.

LAZOS, L., CAPKUN, S., AND POOVENDRAN, R. 2005. Rope: Robust position estimation in wireless sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*.

LAZOS, L. AND POOVENDRAN, R. 2004. Serloc: Secure range-independent localization for wireless sensor networks. In *ACM workshop on Wireless security (ACM WiSe 2004)*. Philadelphia, PA.

LAZOS, L. AND POOVENDRAN, R. 2005. Serloc: Robust localization for wireless sensor networks. *ACM Transactions on Sensor Networks 1*, 1 (August), 73–100.

LEVIS, P., LEE, N., WELSH, M., AND CULLER, D. 2003. TOSSIM: Simulating large wireless sensor networks of TinyOS motes. In *Proceedings of the First ACM Conference onEmbedded Networked Sensor Systems (SenSys 2003)*. 126–137.

LI, Z., TRAPPE, W., ZHANG, Y., AND NATH, B. 2005. Robust statistical methods for securing wireless localization in sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*.

LIU, D., NING, P., AND DU, W. 2004. Attack-resistant location estimation in wireless sensor networks. Tech. Rep. TR-2004-29, North Carolina State University, Department of Computer Science. Revised August 2005.

LIU, D., NING, P., AND DU, W. 2005a. Attack-resistant location estimation in wireless sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*.

LIU, D., NING, P., AND DU, W. 2005b. Detecting malicious beacon nodes for secure location discovery in wireless sensor networks. In *Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS '05)*. 609–619.

NAGPAL, R., SHROBE, H., AND BACHRACH, J. 2003. Organizing a global coordinate system from local information on an ad hoc sensor network. In *IPSN'03*.

NASIPURI, A. AND LI, K. 2002. A directionality based location discovery scheme for wireless sensor networks. In *Proceedings of ACM WSNA'02*. 105–111.

NEWSOME, J. AND SONG, D. 2003. GEM: graph embedding for routing and data-centric storage in sensor networks without geographic information. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys '03)*. 76–88.

NICULESCU, D. AND NATH, B. 2001. Ad hoc positioning system (APS). In *Proceedings of IEEE GLOBECOM '01*.

NICULESCU, D. AND NATH, B. 2003a. Ad hoc positioning system (APS) using AoA. In *Proceedings of IEEE INFOCOM 2003*. 1734–1743.

NICULESCU, D. AND NATH, B. 2003b. DV based positioning in ad hoc networks. In *Journal of Telecommunication Systems*.

OZTURK, C., ZHANG, Y., AND TRAPPE, W. 2004. Source-location privacy in energy-constrained sensor network routing. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks (SASN '04)*. 88–93.

PERRIG, A., SZEWCZYK, R., WEN, V., CULLER, D., AND TYGAR, D. 2001. SPINS: Security protocols for sensor networks. In *Proceedings of Seventh Annual International Conference on Mobile Computing and Networks*. 521–534.

PRZYDATEK, B., SONG, D., AND PERRIG, A. 2003. SIA: Secure information aggregation in sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys '03)*.

RATNASAMY, S., KARP, B., YIN, L., YU, F., ESTRIN, D., GOVINDAN, R., AND SHENKER, S. 2002. GHT: A geographic hash table for data-centric storage. In *Proceedings of 1st ACM International Workshop on Wireless Sensor Networks and Applications*.

RAY, S., UNGRANGSI, R., PELLEGRINI, F. D., TRACHTENBERG, A., AND STAROBINSKI, D. 2003. Robust location detection in emergency sensor networks. In *Proceedings of IEEE INFOCOM 2003*.

SASTRY, N., SHANKAR, U., AND WAGNER, D. 2003. Secure verification of location claims. In *ACM Workshop on Wireless Security*.

SAVVIDES, A., HAN, C., AND SRIVASTAVA, M. 2001. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of ACM MobiCom '01*. 166–179.

SAVVIDES, A., PARK, H., AND SRIVASTAVA, M. 2002. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proceedings of ACM WSNA '02*. 112–121.

SHENKER, S., RATNASAMY, S., KARP, B., GOVINDAN, R., AND ESTRIN, D. 2002. Data-centric storage in sensornets. In *Proceedings of the First ACM Workshop on Hot Topics in Networks*.

YU, Y., GOVINDAN, R., AND ESTRIN, D. 2001. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Tech. Rep. UCLA/CSD-TR-01-0023, UCLA, Department of Computer Science. May.