Attack-Tolerant Localization via Iterative Verification of Locations in Sensor Networks

TAEJOON PARK
Korea Aerospace University
and
KANG G. SHIN
University of Michigan, Ann Arbor

In sensor networks, secure localization—determining sensors' locations in a hostile, untrusted environment—is a challenging, but very important, problem that has not yet been addressed effectively. This paper presents an attack-tolerant localization protocol, called *Verification for Iterative Localization* (VeIL), under which sensors cooperatively safeguard the localization service. By exploiting the high spatiotemporal correlation existing between adjacent nodes, VeIL realizes (a) adaptive management of a profile for normal localization behavior, and (b) distributed detection of false locations advertised by attackers by comparing them against the profile of normal behavior. Our analysis and simulation results show that VeIL achieves high-level tolerance to many critical attacks, and is computationally feasible on resource-limited sensors.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; K.6.m [**Management of Computing and Information Systems**]: Miscellaneous—*Security*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Distributed networks*

General Terms: Algorithms, Reliability, Security

Additional Key Words and Phrases: Anomaly detection, attack-tolerance, localization, recursive least squares, sensor networks

ACM Reference Format:

Park, T. and Shin, K. G. 2008. Attack-tolerant localization via iterative verification of locations in sensor networks. ACM Trans. Embedd. Comput. Syst. 8, 1, Article 2 (December 2008), 24 pages. DOI = 10.1145/1457246.1457248 http://doi.acm.org/10.1145/1457246.1457248

The work reported in this article is supported in part by the ONR and the NRL under Grant No. N00014-04-10726, by the NSF under Grant CCR-0329629, and by Cisco Corporation.

Authors' addresses: T. Park, School of Electronics, Telecommunication and Computer Engineering, Korea Aerospace University, Goyang-city, Gyeonggi-do 412-791, Korea; email: tjpark@kau.ac.kr; K. G. Shin, Real-Time Computing Laboratory (RTCL), Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2121; email: kgshin@eecs.umich.edu.

1. INTRODUCTION

An increasing number of safety- and security-critical applications, such as situation monitoring and facility surveillance, rely on a network of small, inexpensive, battery-powered sensor devices that have limited energy supplies, storage, computation, and communication capacities. Such a sensor network can be used for data acquisition for various applications ranging from physical infrastructure to habitat monitoring. A sensor network is usually built with a large number (thousands or even millions) of resource-limited sensor nodes, each capable of, for example, reading temperature or detecting (part of) an object moving nearby. Sensors cooperate and coordinate with one another to accomplish a higher-level sensing mission, such as accurately measuring and reporting the characteristics of a moving object.

Many applications exist that require each sensor node to be location-aware; for example, each sensor node must be uniquely identified by its location estimate for geographic routing [Jain et al. 2001] in which a source or an intermediate sensor node forwards a packet to one of its neighbors that is closest to the packet's destination. To meet this requirement, various localization schemes [Bulusu et al. 2000; Nicolescu and Nath 2001; He et al. 2003; Hu and Evans 2004; Pathirana et al. 2005; Priyantha et al. 2005; Shang et al. 2003; Ji and Zha 2004; Costa et al. 2005] have been proposed for sensors to determine, with reasonable accuracy, their relative locations within the network coverage area. All these schemes employ location-information-equipped anchors that provide reference locations, and sensor nodes determine their relative locations with respect to the anchors' reference locations. Accordingly, they can successfully accomplish their application/mission only if all participants are benign and strictly follow the localization protocol.

However, sensor networks are usually deployed in a hostile, unattended, and untrusted environment, and hence, they face various critical security attacks from (malicious) compromised nodes. Specifically, an adversary may attempt to fail the localization service by advertising false locations, causing errors in distance measurements, or introducing bogus anchors. Despite its importance, the problem of determining sensor nodes' locations in the presence of attacks has not yet been addressed effectively. Existing solutions either rely on traditional authentication mechanisms [Hu and Evans 2004] or simply use anchor-supplied information [Lazos and Poovendran 2004; Li et al. 2005; Liu et al. 2005], but they fail to completely safeguard the location service due mainly to the noncryptographic nature of attacks, the requirement of unrealistically-powerful anchors, and/or inadequate use of network characteristics.

In this article, we address the problem of "attack-tolerance" in the design of a localization protocol. We consider a large-scale sensor network equipped with only a very small number of less-capable anchors than assumed in the existing schemes, which makes the problem more realistic but challenging. We take an approach to building an attack-tolerant localization protocol, called *verification for iterative localization* (VeIL). This approach is motivated by the fact that a sensor network inherently relies on collective assurance among multiple low-cost sensors to execute high-precision missions, where attack-tolerance is one of such missions.

The heart of VeIL is the use of spatiotemporal correlation among adjacent nodes in the development of anomaly-based attack detection. In essence, VeIL is a cooperative intrusion-detection system tailored to localization, and consists of

—a profile manager that captures, and adaptively tracks, the profile of normal localization behavior; and

—an attack detector that detects and locates attacks by iteratively verifying location announcements via their comparison against the profiled normal profile.

These two building blocks together achieve high-level tolerance to attacks by rejecting any information that exhibits a noticeable deviation from the normal profile, thereby forcing the attacker to weaken the attack strength so as not to be caught, which, in turn, makes it very unlikely for the attacker to fail the localization service. Moreover, our security analysis and simulation results demonstrate the effectiveness and robustness of VeIL in defeating many critical attacks while incurring the processing overhead amenable to resource-poor sensor nodes, and no additional communication overhead.

The rest of the paper is organized as follows. Section 2 reviews the background and related work. Section 3 describes the proposed protocol, VeIL. Section 4 analyzes the security of VeIL while Section 5 evaluates its performance via simulation. Finally, the article concludes with Section 6.

2. BACKGROUND AND RELATED WORK

This section summarizes existing localization methods and describes known attacks on the localization service as well as their countermeasures.

2.1 Localization Algorithms

The localization problem in sensor networks (deployed in a non-malicious environment) is to assign locations to sensors consistently with measured or estimated distances. Techniques for estimating the distance between a pair of communicating nodes are typically based on: (a) received signal strength (RSS) that can be translated into a distance estimate; (b) time of arrival (TOA) and time difference of arrival (TDOA) that use the signal propagation time; and (c) angle of arrival (AOA) that estimates the relative angle between nodes. These techniques are then combined with various signaling methods (e.g., based on RF, ultrasound, or infrared signals) [Youssef and Agrawala 2005; Borriello et al. 2005]. Direct RSS-to-distance conversion, currently supported by motes [Crossbow 2005], becomes inaccurate as the distance increases due mainly to nonuniform signal propagation characteristics and fading/interference effects. To mitigate this estimation error, one may apply averaging or smoothing as in Savarese et al. [2002] and Whitehouse and Culler [2002].

Other than these ranging techniques, range-free schemes have also been proposed to provide cost-effective, coarse-grained localization. In Bulusu et al.

[2000], the location of a sensor is determined as the center of all the anchors it hears. In He et al. [2003], each sensor forms virtual triangular regions among the anchors of interest, determines in which regions it resides based on its neighbors' measurements, and finally calculates the overlap of those regions. These schemes typically require very high anchor density and long anchors' radio ranges as each sensor has to hear from as many anchors as possible.

It is preferable to provide localization capability even when there are only a very small number of anchors in the network. In the hop-count-based localization [Nicolescu and Nath 2001], each sensor determines the minimum hop-counts to anchors by running a distance vector algorithm, and computes physical distances by multiplying them to the average per-hop distance. This scheme, unfortunately, yields poor localization accuracy. Other approaches [Hu and Evans 2004; Pathirana et al. 2005; Priyantha et al. 2005] employ mobile anchors to meet the requirements of both low anchor density and high accuracy of distance estimation, but suffer a large latency.

The highest localization accuracy (in terms of minimizing the difference between assigned and real locations) can be achieved by utilizing multidimensional scaling (MDS), widely used in mathematical psychology, economics, sociology and machine learning communities for modeling proximity relations. MDS-MAP [Shang et al. 2003] constructs network-wide connectivity information in the form of a matrix of all possible distance estimates (in hop-counts), and then applies MDS to derive sensors' locations that fit well those estimated distances. However, it must rely on a central processing node that collects all distance estimates and computes location assignments, significantly degrading scalability due mainly to its high communication overhead.

The MDS technique is flexible enough to find consistent location assignments even when a limited set of distance estimates are available. In such a case, one may apply various *iterative MDS* techniques, for example, those reported in [Basalaj 2001]. Ji and Zha [2004] took this approach to develop a distributed localization scheme by applying MDS iteratively to build a local map of locations for each group of adjacent sensors and then combining these maps together to obtain a global location map. This scheme requires the computation of eigendecomposition per iteration that takes $\mathcal{O}(n^3)$ time for a group of n sensors. Costa et al. [2005, 2006] also developed a distributed, iterative MDS scheme that (a) relies solely on distance measurements between neighboring sensors and (b) is computationally less demanding than that of Ji and Zha [2004].

2.2 Security Attacks

Attacks on a sensor network can be classified as: (a) physical attacks on sensor devices (e.g., destroying, capturing, reverse-engineering, reprogramming and/or cloning sensors), (b) service-disruption attacks on routing, localization, and time synchronization, (c) data attacks (e.g., traffic capture, replaying and spoofing), (d) resource-consumption and denial-of-service (DoS) attacks that diminish or exhaust the sensors' capacity/energy to perform their normal function, and (e) sybil attacks [Douceur 2002] by which a single malicious sensor device claims/presents multiple IDs (locations) to control a substantial fraction of the ID space which, in turn, makes it easier to mount other attacks. These attacks are generally coupled together; for example, a small number of compromised sensors created via a physical attack may serve as zombies for many serious attacks, such as initiating DoS or sybil attacks, disrupting network services, and so on. Tamper-proofing techniques, such as those of Park and Shin [2005], can be used to detect/reject compromised sensors.

In this article, we focus on localization-specific attacks. Possible attacks [Hu and Evans 2004; Li et al. 2005; Capkun and Hubaux 2004] on the localization service include:

—sensor displacement or removal;
—distance enlargement/reduction, such as via jamming, adjustment of transmission power, or placement of obstacles interfering with direct paths;
—announcement of false locations, distances or hop-counts;
—message modification or replaying;
—wormhole attacks that create hidden links between remote (compromised) sensors to be used for replaying messages or altering distance measurements or hop-counts; and
—deployment of bogus anchors that propagate false reference location information.

All these attacks try to propagate wrong information about locations of, or distances to, the sensors (or anchors) under the adversary's control in an attempt to disrupt the localization service.

2.3 Countermeasures against Attacks on Localization Service

As mentioned earlier, determining sensors' locations in an untrusted environment, is a challenging problem that has not yet been fully studied. Like other security applications, one may want to authenticate all the messages to protect the network against attacks (targeting at data traffic). For this purpose, as discussed in Hu and Evans [2004], one may attempt to use digital signatures or μTesla [Perrig et al. 2001] together with key predeployment schemes [Eschenauer and Gligor 2002; Chan et al. 2003]. However, the former suffers a high computational overhead while the latter suffers a large authentication latency, and, more importantly, many of the above-mentioned localization-targeted attacks are noncryptographic in nature, making these authentication-based solutions highly unlikely to succeed.

The method proposed by Lazos and Poovendran [2004] is conceptually similar to He et al. [2003] in that each sensor hears directly from multiple anchors, identifies a region it resides in, and determines its location as the center of the region. In this scheme, the security against chosen attacks is preserved if the anchors are trusted and cannot be compromised by the adversary. However, its main drawback is the requirement of a large number of specialized anchors equipped with directional/sectored antennae and capable of high power transmission.

Recently, statistical approaches have been proposed [Li et al. 2005; Liu et al. 2005]. Li et al. [2005] presented an attack-tolerance mechanism for triangulation-based localization in which each sensor applies the least median squares algorithm on the distance estimates to anchors in order to mitigate the effect of attacks. Liu et al. [2005] also use a collection of anchors' reference locations associated with estimated distances, and apply the mean square error criterion to identify and discard malicious location references. Unfortunately, these methods invite attacks from relaying sensors and require a significant amount of redundant location/distance information from anchors, incurring a high network overhead to achieve a reasonable degree of robustness against attacks. These drawbacks mainly come from the fact that they do not fully extract/utilize the available information and ignore the relationship among sensors' locations.

Although not directly applicable to localization, Sastry et al. [2003] and Waters and Felten [2003] developed algorithms to verify the distance or location claims of a node (e.g., to ensure the node to be within a certain region). They rely on a challenge-response protocol that measures the round-trip time between a verifier and the node, and then translate the elapsed time into distance. Another way [Capkun and Hubaux 2004] is to check if the node's location falls within a triangular region formed by three trusted verifiers. These algorithms use centralized trusted servers, and hence, can be used as local defense mechanisms against distance-reduction attacks, but not as global, general-purpose solutions.

3. THE PROPOSED PROTOCOL

It is almost impossible to completely prevent all possible attacks in any form of systems including sensor networks, so we should instead make a system "attack-tolerant." To achieve this goal, we propose a localization protocol, called verification for iterative localization (VeIL), that

(1) tolerates attacks (and faults) by malicious (misbehaving) devices,
(2) incurs a small processing overhead without any communication overhead,
(3) preserves compatibility with other services like the authentication framework, and
(4) achieves high-localization accuracy and efficiency.

Described below are the network and threat models, the proposed countermeasures, the underlying localization algorithm, and the details of VeIL.

3.1 The Network Model

The sensor network under consideration consists of sparsely-deployed, less-capable, static anchor nodes and a large number of sensor devices.[1] This is

[1]Note this relaxation significantly lowers the deployment cost because anchor nodes do not necessarily cover the entire network area (e.g., via the high-power transmission). Note also that the more anchors a network is equipped with, the better performance VeIL achieves. See Section 4.3 for the effects of mobile anchors on VeIL.

a realistic deployment scenario in that the sensor network is inherently an infrastructure-less network in which sensors autonomously organize themselves into a connected structure, and hence, it is desirable to minimize the dependency of localization on infrastructure nodes, such as anchors. In this environment, it is not uncommon that a sensor cannot directly hear from any of the anchors, necessitating collective assurances among sensors to reach network-wide consistent location assignments.

We choose the MDS-based iterative localization algorithm [Ji and Zha 2004; Costa et al. 2005] as our underlying localization scheme in which each and every sensor keeps refining its location estimates based on location announcements from, and distance measurements with, its direct neighbors. To this end, two nodes within each other's transmission range establish a mutual neighborhood relationship. Any of the ranging techniques (RSS, TOA, TDOA, and AOA) described earlier can be used to estimate distances between direct neighbors. Sensors may optionally use location verification protocols [Sastry et al. 2003; Waters and Felten 2003] to ensure their neighbors are really within their communication range.

3.2 The Threat Model

Anchors are assumed to be trusted entities, that is, the reference locations provided by them are trustworthy and cannot be spoofed by the adversary. Accordingly, each sensor can authenticate the reference locations to confirm that they are indeed from genuine anchors. Note, however, that VeIL is resilient against attacks from compromised anchors. The effects of malicious anchors will be discussed in Section 4.3.

By contrast, sensors (regardless of their physical proximity to the anchors) can be physically compromised or tampered with by the adversary at any time. Therefore, the localization takes place in the presence of malicious/compromised devices (the number of which is less than a majority[2] of that of network nodes in a given area, either the entire network or the local region) disguised as normal participants who will do their best to disrupt the localization service by mounting the attacks described in Section 2.2. Note that no algorithm can succeed if a majority of nodes lie about their locations, and hence, this assumption does not limit the effectiveness of VeIL.

3.3 The Proposed Approach

To maximize both attack-tolerance and localization-accuracy, we exploit the spatiotemporal correlation among neighboring nodes' locations to determine if a malicious node claims/announces false locations. Basically, the adversary must be aggressive enough to disrupt the localization service. However, if a malicious device advertises arbitrary locations that deviate significantly from what the protocol expects, its neighbors, if a majority of them are well-behaving, would easily detect the discrepancy via cooperative location validation to blacklist/block the misbehaving sensor from participating in the localization service.

[2]A simple, or two-thirds majority in case of Byzantine faults.

So, the malicious sensor must risk getting caught if its falsification deviates too much from its normal location because its unusual distances to its neighbors make it conspicuous during the localization process. On the other hand, the false locations with small perturbations wouldn't do any harm, since the effects of small perturbations would easily be canceled out.

We take an anomaly detection approach [Zhang and Lee 2000; Mishra et al. 2004] to realize this idea. That is, each sensor maintains, and adaptively updates, a baseline profile of the normal localization behavior based on past announcements of all its neighbors. Then, upon reception of new announcements, it compares them with this normal profile, and if a noticeable deviation is found, decides on the presence of a possibly adversarial behavior and takes an action to locate the malicious or misbehaving sensor. Consequently, VeIL consists of the following two building blocks that closely interact with each other:

—**a profile manager** (described in Section 3.5) that constructs and maintains the compact profile of normal localization behavior; and

—**an attack detector** (described in Section 3.6) that detects, locates, and rejects false location announcements, as well as updates the normal profile.

VeIL is essentially a cooperative intrusion detection mechanism tailored to localization, in which each and every sensor checks if the location/distance announcement from each of its neighbors is "abnormal," and, if so, removes the sensor from the rest of the localization process. Unlike the other schemes that rely solely on anchors, VeIL uses peer sensors as active information sources that provide distances information and incremental location updates, thereby maximizing the attack-detection capability.

3.4 The Underlying Localization Algorithm

Let the index s refer to the sensor performing localization, and n_s denote the number of s's direct (one-hop) neighbors. The (local) indices, $i = 1, \dots, n_s$, are assigned to s's neighbors, each of which may or may not be an anchor node. There exists one-to-one correspondence between the index and ID of a sensor. The distance estimate between s and i is denoted by $\delta_{s,i}$. Also, let $w_{s,i}$ denote a weight assigned between s and i, the value of which is either binary (1 if $\delta_{s,i}$ is known; 0 otherwise) [Ji and Zha 2004] or adaptively chosen [Costa et al. 2006]. We define the *individual cost* between s and i in the k^{th} iteration as

$$c_{s,i}(k) = w_{s,i} \left[\delta_{s,i} - \|\mathbf{x}_s(k) - \mathbf{x}_i(k)\| \right]^2 \tag{1}$$

where $\mathbf{x}_s(k)$ and $\mathbf{x}_i(k)$ are $p \times 1$ coordinate vectors ($p = 2$ or 3 for two- or three-dimensional coordinates, respectively) representing estimated locations of s and i at iteration k (≥ 0), respectively, and $\|\mathbf{x}\|$ denotes the Euclidean norm of the vector $\mathbf{x}$. Then, the *local cost* of sensor s, $c_{s,0}(k)$, at iteration k is computed by summing up all individual costs:

$$c_{s,0}(k) = \sum_{i=1}^{n_s} c_{s,i}(k). \tag{2}$$

The localization algorithm searches s's true location by iteratively minimizing $c_{s,0}(k)$. The entire localization process works as follows.

—Initially, all sensors in the network randomly choose their initial location estimates while the anchors use their own fixed reference locations, that is, $\mathbf{x}_s(0)$ and $\mathbf{x}_i(0)$ are the initial locations of s and i.

—At iteration k ($\geq$0), s refines its location estimate, $\mathbf{x}_s(k+1)$, by processing $\{\delta_{s,i}, \mathbf{x}_i(k)\}_{i=1}^{n_s}$ with the updated formula in Bulusu et al. [2000], Ji and Zha [2004] and Costa et al. [2006], then exchanges the new locations with all its neighbors.

—s terminates the algorithm if the location estimate gets stabilized (i.e., $c_{s,0}(k) - c_{s,0}(k+1) < \epsilon$); otherwise, repeat the process at the next iteration.

The communication overhead incurred by exchanging location estimates is small because each sensor may simply broadcast this information as part of the beaconing process (that periodically exchanges BEACON packets to refresh sensors' neighbor-lists). Also, note that beaconing is one of the basic operations a sensor must execute throughout its lifetime. Therefore, s may keep on fine-tuning its location estimate after terminating the above algorithm based on the BEACON packets exchanged. Our proposed protocol can be used to verify BEACON packets, thus serving as an online guard mechanism against attacks targeting at sensors' locations, such as sybil attacks.

3.5 Construction of Normal Profiles

We want to construct a profile of s for the normal localization behavior, based solely on the information collected by s during the localization. That is, in the k^{th} iteration, s has been processing $\{\mathbf{x}_i(t)\}_{t=1}^{k}$ using Eq. (1) to compute $k \times 1$ vectors,

$$\mathbf{c}_{s,i}(k) = [\, c_{s,i}(k), \dots, c_{s,i}(1)\,]^T, \quad 1 \leq i \leq n_s. \tag{3}$$

Then, a $k \times 1$ local cost vector of s, $\mathbf{c}_{s,0}(k)$, is given by

$$\mathbf{c}_{s,0}(k) = [\, c_{s,0}(k), \dots, c_{s,0}(1)\,]^T = \sum_{i=1}^{n_s} \mathbf{c}_{s,i}(k). \tag{4}$$

Figure 1 plots typical $\mathbf{c}_{s,i}(k)$ values when $i = 1, \dots, 12$ and $k = 30$. From this figure, we observe the following two facts:

—each $\mathbf{c}_{s,i}(k)$ exhibits strong temporal correlation; and

—$\mathbf{c}_{s,0}(k)$ is strictly decreasing as the iteration progresses, due to the spatial correlation among neighbors.

We can, therefore, derive as compact a description of the normal profile as possible by removing this redundancy. We describe below how to exploit this property *optimally* (in the sense of achieving the highest attack-resolution, i.e., if any other scheme can resolve an attack, so can VeIL).

3.5.1 *Problem Formulation.* Our problem is cast into the design of an adaptive transversal filter bank (of s) that consists of n_s filters, each with M taps, thus bookkeeping the localization history of all s's neighbors for the past M

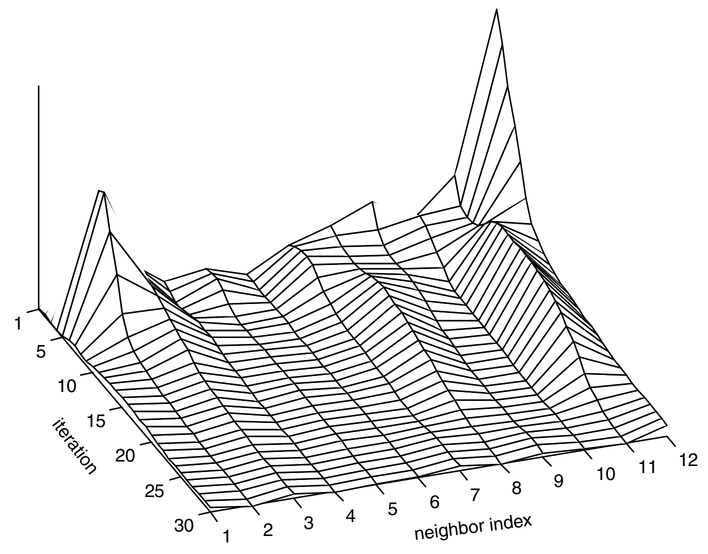


Fig. 1. Example plot of $\mathbf{c}_{s,i}(k)$ for $1 \leq i \leq 12$ and $k = 30$.

iterations. We first formulate a *least squares prediction* problem as follows:

$$\left.\begin{array}{rcl} \hat{c}_{s,1}(t) & = & \mathbf{h}^T_{s,1}(k)\,\mathbf{c}_{s,1}(t-1;M) \\ & \cdots & \\ \hat{c}_{s,n_s}(t) & = & \mathbf{h}^T_{s,n_s}(k)\,\mathbf{c}_{s,n_s}(t-1;M) \end{array}\right., \; M < t \leq k, \tag{5}$$

where $k \geq M+1$ and $\mathbf{h}_{s,i}(k)$ is the $M \times 1$ filter-weight vector for neighbor i at iteration k defined by

$$\mathbf{h}_{s,i}(k) = [\, h_{s,i1}(k), \ldots, h_{s,iM}(k)\,]^T \tag{6}$$

and $\mathbf{c}_{s,i}(t-1;M)$ is the $M \times 1$ past individual cost vector for i given by

$$\mathbf{c}_{s,i}(t-1;M) = [\, c_{s,i}(t-1), \ldots, c_{s,i}(t-M)\,]^T. \tag{7}$$

Our objective is to find, at iteration k, estimators $\{\hat{\mathbf{h}}_{s,i}(k)\}_{i=1}^{n_s}$, each of which minimizes the sum of squared errors (SSE):

$$SSE_{s,i}(k) = \sum_{t=M+1}^{k} \lambda^{k-t}\,|\, c_{s,i}(t) - \mathbf{h}^T_{s,i}(k)\mathbf{c}_{s,i}(t-1;M)\,|^2 \tag{8}$$

where λ ($\leq$1) is an exponential forgetting factor. The smaller the value of λ, the higher the weight on the more recent information.

3.5.2 *Recursive Least Squares Algorithm.* We apply the method of recursive least squares (RLS) [Haykin 1991] to develop a recursive algorithm that updates the filter-weight vectors $\{\hat{\mathbf{h}}_{s,i}(k)\}_{i=1}^{n_s}$ upon reception of $\{\mathbf{x}_i(k)\}_{i=1}^{n_s}$ (translated into $\{c_{s,i}(k)\}_{i=1}^{n_s}$), given $\{\hat{\mathbf{h}}_{s,i}(k-1)\}_{i=1}^{n_s}$, where $k \geq M+1$. The RLS algorithm first calculates, for each i, a priori prediction error based on old filter-weight

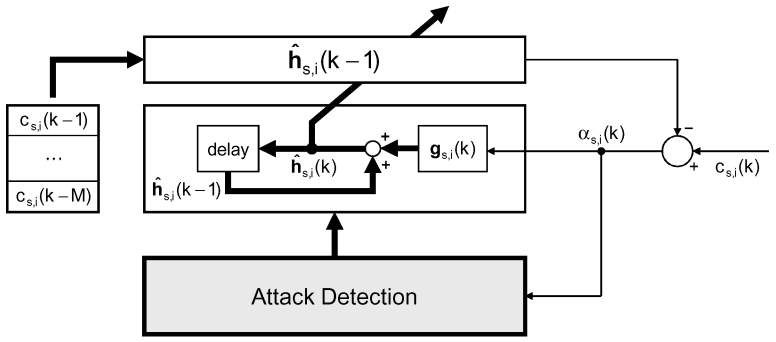


Fig. 2. The VeIL architecture.

estimates at iteration k, as follows:

$$\alpha_{s,i}(k) = c_{s,i}(k) - \hat{\mathbf{h}}_{s,i}^T(k-1)\mathbf{c}_{s,i}(k-1;M). \tag{9}$$

The filter-weight vector is then updated as

$$\hat{\mathbf{h}}_{s,i}(k) = \hat{\mathbf{h}}_{s,i}(k-1) + \alpha_{s,i}(k)\,\mathbf{g}_{s,i}(k) \tag{10}$$

where $\hat{\mathbf{h}}_{s,i}(M) = \mathbf{0}$ and an $M \times 1$ gain vector $\mathbf{g}_{s,i}(k)$ is computed by

$$\mathbf{g}_{s,i}(k) = \frac{\mathbf{P}_{s,i}(k-1)\mathbf{c}_{s,i}(k-1;M)}{\lambda + \mathbf{c}_{s,i}^T(k-1;M)\mathbf{P}_{s,i}(k-1)\mathbf{c}_{s,i}(k-1;M)} \tag{11}$$

$\mathbf{P}_{s,i}(k)$ is an $M \times M$ inverse correlation matrix, initialized to

$$\mathbf{P}_{s,i}(M) = \rho^{-1}\mathbf{I} \tag{12}$$

with a small positive ρ, and recursively updated by

$$\mathbf{P}_{s,i}(k) \ = \lambda^{-1}\mathbf{P}_{s,i}(k-1) - \lambda^{-1}\mathbf{g}_{s,i}(k)\mathbf{c}_{s,i}^T(k-1;M)\,\mathbf{P}_{s,i}(k-1). \tag{13}$$

For details of how the RLS algorithm works, see [Haykin 1991].[3]

3.5.3 *Profile Manager.* Figure 2 shows the architecture of profile manager constructed based on the above RLS algorithm. The profile manager of s includes n_s RLS filters, each storing M past individual costs, $\mathbf{c}_{s,i}(k-1;M)$, and M filter-weights, $\hat{\mathbf{h}}_{s,i}(k)$, that are adaptively and recursively updated using Eqs. (9)–(13) starting at $k = M+1$. With this architecture, the entire localization history of s, up to the k^{th} iteration, is fully captured in $n_s \times M$ filter-weights, $\{\hat{\mathbf{h}}_{s,i}(k)\}_{i=1}^{n_s}$, that constitute a normal profile at iteration k. $\{\hat{\mathbf{h}}_{s,i}(k)\}_{i=1}^{n_s}$ is updated only if $\{\mathbf{x}_i(k)\}_{i=1}^{n_s}$ is verified to be genuine according to the procedure described in Section 3.6, thereby ensuring the normal profile to be updated from attack-free data.

[3]RLS achieves the optimum fit between estimation and real measurements by recursively updating the filter-weight vectors to minimize the sum of squares of prediction errors. The benefits of using the RLS algorithm is that it does not require inversion of large matrices, hence being very efficient and reducing the computational requirement.

The computational requirement at each iteration is $\mathcal{O}(n_s \cdot M^2)$, implying that the CPU and energy overheads at each sensor depend on the choice of M. It is, therefore, important to configure the network with a proper M value. Clearly, there exists a tradeoff between the accuracy of normal profile and the overhead of profile management (i.e., the larger value of M yields a more accurate profile at the expense of higher overhead). Moreover, a smaller value of M should be preferred as far as they maintain a reasonable level of accuracy, due mainly to their resource constraints.

To determine an optimal value of M in terms of balancing the accuracy and overhead, one may use either the information-theoretic criterion (AIC) or the minimum description length (MDL) criterion [Haykin 1991]. But, in this paper, we took an approach to find the best M value through simulation instead of deriving the optimality condition. As shown in Section 5.3, the prediction accuracy of the profile manager remains almost the same for the M values of 3 or higher, thanks to the high temporal correlation. Hence, we conclude the best choice of M value is 3 in that it minimizes the computational overhead without sacrificing the prediction accuracy.

Finally, it is possible to use the Kalman filter instead of RLS, in the sense that the above RLS algorithm (which is a tailored solution to our prediction problem) is indeed a special case (i.e., sharing the same mathematical structure) of the Kalman filter. The application of Kalman filtering to our prediction problem given by Equation (5) will yield Equations (9) through (13) to be replaced with the Kalman filter recursions. The computational complexity in using this general-purpose filter is acceptable for small M values.

3.6 Detection of Attacks

We aim to design an attack detector that detects, or mitigates the effects of, attacks presented in Section 2.2. With these attacks, a malicious sensor i attempts to have a falsified $\mathbf{x}_i(k)$ accepted by s so that $c_{s,i}(k)$ can be boosted to a large value. This will cause $\mathbf{x}_s(k+1)$ to have a large deviation from its desired value, making it impossible, or at least take a very long time, for s to determine its true location. We describe below how VeIL defends a network against this threat, and then qualitatively analyze its attack-detection capability.

3.6.1 *Proposed Detection Scheme.*

First of all, possible attacks during the first M iterations can be found easily because the localization starts with high individual costs even in an attack-free environment due to the randomly-assigned initial locations while the cost values gets smaller as the iteration goes on. That is, the profile manager of s initially keeps populating its buffer with cost values and then activates the prediction mechanism at the $(M+1)$-th iteration when the buffer is filled. This means the malicious neighbor i cannot arbitrarily falsify its announcements (from iteration $M+1$ on) because the announced cost must be consistent with $\hat{c}_{s,i}(M+1)$ $(= \hat{\mathbf{h}}^T_{s,i}(M)\mathbf{c}_{s,i}(M;M))$. Consequently, if i made false announcements during the first M iterations, it has to keep misbehaving after this initial period, but doing so will get caught because its cost value will soon become conspicuous among s's neighbors. Thus, the malicious neighbors won't be able to confuse/disrupt the synchronization process by attacking the initial M iterations.[4]

We, therefore, propose an incremental detection scheme activated at iteration $M+1$, in which every sensor verifies the trustworthiness of incremental location updates, $\{\mathbf{x}_i(k)\}_{i=1}^{n_s}$, from its neighbors by comparing them with the normal profile built during the $(k-1)^{\text{th}}$ iteration. VeIL achieves this comparison easily by evaluating Eq. (9) for $i = 1, \ldots, n_s$. Clearly, $\alpha_{s,i}(k)$, $1 \leq i \leq n_s$, quantifies the difference of i's new announcement from its value predicted from the most-recent profile, and hence, s should suspect i to be malicious if $\alpha_{s,i}(k)$ exceeds a certain threshold. Accordingly, for each i, s decides $\mathbf{x}_i(k)$ to be disruptive/harmful to the location service if

$$|\alpha_{s,i}(k)| \geq \eta_t \cdot \max\{c_{s,i}(k),\ c_{\min}\} \tag{14}$$

where η_t (≤ 1) is a preconfigured network-wide threshold for detecting anomalies, the optimal value of which can be determined via statistical methods like an F-test, and $c_{\min}$ is the minimum individual cost, below which the prediction error becomes negligible because $\|\mathbf{x}_s(k) - \mathbf{x}_i(k)\|$ falls well within $\delta_{s,i}$.

We develop another detection rule by utilizing the fact that $c_{s,0}(k)$ must decrease with the iteration count if there is no attacker. Hence, every sensor should monitor if this condition is violated; in the k^{th} iteration, s checks if

$$c_{s,0}(k) \geq \eta_0(k) \cdot c_{s,0}(k-1), \tag{15}$$

where $\eta_0(k)$ (≤ 1) is a threshold to verify the acceptability of local cost whose value is determined by the choice of the underlying localization algorithm. Equation (15) will be triggered if $c_{s,0}(k)$ does not decay as expected because of the domination by a few anomalous individual costs from malicious nodes.

Our proposed detection scheme can defeat (or at lease stress) the intelligent attacker i that persistently (e.g., from the beginning) gives incorrect information based on the knowledge of normal profiles, because i is forced to weaken the attack strength, that is, in terms of the degree of incorrectness, to make it consistent with Equation (14) as long as there are less than a majority of malicious nodes within the local region of concern, but doing so cannot disrupt the localization service at all. Otherwise, i may try to fool the construction of normal profiles during the initial learning period in such a way that the neighbors predict wrong (thus high) cost values for i's announcements, thereby allowing the injection of fake locations/distances (i.e., consistent locally, but inconsistent globally) without getting caught by Equation (14). However, this attack will soon trigger Equation (15) because the cost from i, if obeying Equation (14), will become conspicuous among the costs of other neighbors. This shows that evading both Equations (14) and (15) is very difficult even for the determined attackers.

3.6.2 *Attack Detector.* Figure 2 shows how the VeIL's attack detector interacts with its profile manager. First, both the profile manager and the attack detector are triggered by $\alpha_{s,i}(k)$'s computed at iteration k. The attack detector then processes $\alpha_{s,i}(k)$'s using Equations (14) and (15) to determine if there

[4]Section 4.1 also discusses how VeIL defeats this attack.

exist anomalies in the location announcements, and if so, identifies which of the neighbors caused the anomalies. Finally, this information is fed to the profile manager to reject (announcements from) those neighbors.

The attack detection hinges on a blacklisting scheme: s detects, for each iteration, if there exist anomalies in its neighbors' location announcements, and if so, identifies which of the neighbors caused the anomalies. That is, s decides on the presence of anomalies if either of Equations (14) and (15) are satisfied. s then blacklists a neighbor if it has been caught more than N_B times out of B iterations. The ratio N_B/B (≤ 1) is a design parameter; the choice of N_B/B close to 0 implies an overly-conservative mode of operation, while the opposite ($N_B/B \to 1$) means lenience against attacks. Note it is not so effective for the adversary to exploit this design choice because he still has to weaken the attack strength as iteration goes on. For the purpose of blacklisting, s keeps track of the number of anomalous location announcements from i in blacklist_counter(i). The development of soft blacklisting based on weighted MDS is our future work. (See Section 6 for details.)

3.6.3 *Attack-Detection Capability.* VeIL must be able to amplify the prediction errors caused by false location announcements for M consecutive iterations, thus detecting anomalies with high accuracy (each of which presents multiple chances to be caught). In other words, sensor i's multiple false announcements within M consecutive iterations will cause VeIL to continuously produce high $\alpha_{s,i}(\cdot)$ values, making it highly likely to detect the misbehaving sensor i. On the other hand, sporadic attacks will be detected by the N_B/B scheme mentioned earlier. Weakening the attack frequency lower than this imposes no threat to the localization service.

Let us consider the case where a malicious sensor i mounted an attack to force s to compute $c_{s,i}(k)$ that differs from the expected cost $c^*_{s,i}(k)$ by Δ, that is, $c_{s,i}(k) = c^*_{s,i}(k) + \Delta$.[5] This will increase $\alpha_{s,i}(k)$ by Δ, because $\alpha_{s,i}(k) = \alpha^*_{s,i}(k) + \Delta$ from Equation (9). Hence, the deviation of the prediction error at iteration k is proportional to Δ. By contrast, the prediction error gets amplified rapidly at iteration $k+1$ for the following reason. From Equation (10), the filter-weights are updated as $\hat{\mathbf{h}}_{s,i}(k) = \hat{\mathbf{h}}^*_{s,i}(k) + \Delta \cdot \mathbf{g}_{s,i}(k)$. Then, $\alpha_{s,i}(k+1)$ can be rewritten as a function of Δ:

$$\alpha_{s,i}(k+1) = \alpha^*_{s,i}(k+1) - \Delta^2 \cdot g_{s,i1}(k) - \Delta \cdot \left[\hat{h}_{s,i1}(k) + \mathbf{g}^T_{s,i}(k)\mathbf{c}_{s,i}(k;M)\right] \quad (16)$$

where $g_{s,i1}(k)$ is the first element of $\mathbf{g}_{s,i}(k)$. Therefore, the perturbation Δ introduced at iteration k results in a very large amount of prediction error in the next iteration. Moreover, this magnification of prediction errors will persist for the period of M consecutive iterations, during which the false cost $c_{s,i}(k)$ remains cached inside the profile manager, thus making it very difficult for the attacker to evade VeIL. This qualitatively illustrates the highest level of VeIL's attack-detection capability. Note that its quantitative evaluation via simulation is given in Section 5.4. Note also that the choice of η_t is not a critical factor

[5] Δ can be viewed as the attack strength in that a larger Δ yields a larger deviation of locations from their true values.

```
Initialization:
    $\hat{\mathbf{h}}_{s,i}(M) = \mathbf{0}$;
    $\mathbf{P}_{s,i}(M) = \rho^{-1}\mathbf{I}$;
    randomly choose and announce $\mathbf{x}_s(0)$;

Iteration:
    // Execute localization until its convergence
    for ($k = 0$; $c_{s,0}(k-1) - c_{s,0}(k) < \epsilon$; $k$++)
        receive $\mathbf{x}_i(k)$ from all $i$;
        compute $c_{s,i}(k)$, $\forall i$, and $c_{s,0}(k)$ using Eqs. (1) and (2);
        // Activate VeIL at iteration $M+1$
        if $k \geq M+1$,
            for $i = 1$ to $n_s$,
                compute $\alpha_{s,i}(k)$ using Eq. (9);
                if $c_{s,0}(k) \geq \eta_0(k) \cdot c_{s,0}(k-1)$ or
                    $|\alpha_{s,i}(k)| \geq \eta_t \cdot \max\{c_{s,i}(k), c_{\min}\}$,
                    $c_{s,i}(k) = \hat{\mathbf{h}}^T_{s,i}(k-1)\, \mathbf{c}_{s,i}(k-1; M)$;
                    if ++ blacklist_counter($i$) $\geq N_B$,
                        blacklist $i$;
                if $i$ blacklisted,
                    deactivate $\hat{\mathbf{h}}_{s,i}(k)$;
                else,
                    update $\hat{\mathbf{h}}_{s,i}(k)$, $\mathbf{g}_{s,i}(k)$ and $\mathbf{P}_{s,i}(k)$;
        // Execute the location-update algorithm
        update and announce $\mathbf{x}_s(k+1)$;
```

Fig. 3. Pseudocode for VeIL at sensor s.

thanks to VeIL's attack-amplification property, with which the value of $|\alpha_{s,i}(k)|$ is boosted to a very large value as demonstrated in Section 5.4.

3.7 Protocol Description

Figure 3 provides the pseudocode of our proposed localization protocol executed at sensor s. (For simplicity, B is set to be unbounded in the pseudocode.) It integrates the operations for the attack detector, the profile manager, and the underlying localization algorithm. The localization at sensor s starts by initializing the profile manager and randomly choosing $\mathbf{x}_s(0)$ followed by announcement of the initial location to its neighbors. Then, s executes up to M iterations with VeIL disabled, activates VeIL at iteration $M+1$, and finally, performs the rest of localization until its convergence.

There are two response mechanisms to false location announcements. First, the false locations (at iteration k) must be excluded from (a) the computation of s's next location estimate $\mathbf{x}_s(k+1)$, and (b) the recursion for the profile manager by letting $c_{s,i}(k) = \hat{\mathbf{h}}^T_{s,i}(k-1)\mathbf{c}_{s,i}(k-1; M)$. Second, malicious neighbor i must be removed from the future localization process (e.g., by deactivating $\hat{\mathbf{h}}_{s,i}(k)$) if it had been caught more than N_B times. To do this, the blacklist_counter(i) is incremented whenever i's announcement is found to be "false."

4. SECURITY ANALYSIS

We classified the attacks in Section 2.2 into three types: (a) location-targeted attacks, (b) distance-targeted attacks, and (c) anchor-targeted attacks. We describe how VeIL counters each of these attacks.

4.1 Defense against Location-Targeted Attacks

The attacker may influence, and cause a significant bias in, the localization process by providing false location information. This threat can be caused by (a) physical attacks that compromise sensors and then deploy them, and (b) wormhole attacks that create hidden links between malicious devices, both of which will then be used to tunnel/replay/modify/create messages carrying location information in an attempt to confuse chosen sensors. Traditional authentication-based countermeasures will likely fail since it is difficult to keep cryptographic keys secret under these attacks. Likewise, any protocol (including the one based on hop-counts) that uses sensors as relays, is vulnerable to these attacks.

By contrast, VeIL is very robust to these location-targeted attacks as it hinges on highly correlated localization behaviors of sensors that update location estimates toward their true locations. Since every sensor keeps refining its location estimate such that the aggregate differences in location information received from its neighbors fit better with measured distances, the next location estimate of a sensor can be predicted, for the most part, from its past localization history. Actually, VeIL makes "optimal" prediction using the least squares formulation and the RLS method. Under this prediction framework, any location announcements that deviate significantly from the corresponding prediction are highly likely from attackers.

An attacker may attempt to break VeIL in several ways. First, he may judiciously adjust the strength of perturbation to make the cost just below the detection threshold η_t. Since the location differences converge to the distance measurements during the localization, such an attempt will place great stress on the attacker into steadily lowering the attack strength not to be caught by VeIL. However, by doing so, the attacker cannot succeed in his mission to fail the localization process. Second, the attacker may try to inject false location information from the beginning, and then keep telling a lie that does not deviate from the original lie. However, such an act will soon be caught if less than a majority of neighbors are lying because the individual cost from the malicious node will become conspicuous among the costs of all the neighbors. Moreover, it is extremely difficult for the attacker to fool all the neighbors without physically compromising them. Third, the attacker may mount sybil attacks. But sybil attacks are ineffective under VeIL, because fictitious locations can only take values agreeing with the corresponding distance measurements to evade VeIL, forcing all the falsified locations to be close to one another. As a result, the only effective way to evade VeIL "locally" is to compromise at least a majority of the sensors within the local region.[6] However, this attack would not be effective as it only disrupts nodes inside or near the affected region, and hence, the rest of the network nodes still compute correct locations. This proves the highest attack-tolerance of VeIL on location-targeted attacks.

[6]Note that this attack has the same effect as that of the anchor-targeted attacks in Section 4.3.

4.2 Defense against Distance-Targeted Attacks

The distance measurements/estimates can be altered by distance enlargement/reduction attacks (e.g., via jamming, physical obstacles and the transmission power control), wormhole attacks on distance or hop-count information, etc. In general, any attempt by a malicious sensor i to modify the distance to its neighbor s makes only a one n_s-th contribution to s's location estimate, since s determines its location with respect to the locations/distances of all n_s neighbors. s's decision is then fed back to the others, canceling the errors from i. So, the impact of the distance-targeted attacks of a single malicious node on sensors in its proximity is small, and smaller on farther-away sensors. Thus, despite its possible undesirable local effects, VeIL will not distort the network-wide connectivity, making it robust to distance-targeted attacks.

Specifically, jamming a local area (or placing obstacles or wormholes) blocks sensors inside (or nearby) the area from participating in the localization. But, the unaffected sensors can still maintain consistent network connectivity among themselves, although their location estimates may differ from real locations (i.e., the closer to the jammed area, the larger the deviation). Moreover, as soon as the jamming is over, the sensors will start adjusting their locations in the course of neighborhood management via BEACON packets. Note that VeIL also plays a role of verifying BEACON packets. Besides, a malicious sensor i may amplify its transmission power in order to proliferate bogus information as well as to cause smaller-distance measurements, but i is more likely to be caught by VeIL because in such a case there will be more neighbors watching on it.

4.3 Defense against Anchor-Targeted Attacks

This type of attacks can be considered as a special case of location-targeted attacks in that malicious/bogus anchors will provide false information on location references. Typically, the effects of location-targeted attacks on anchors become much more serious than those on nonanchor nodes because they will eventually corrupt the entire network. For this reason, anchors are assumed to be trusted entities. However, in reality, anchors can be compromised (by determined attackers) no matter how well protected they are, and, if that happens, it is impossible for sensors to determine their true locations. Also, compromising all (or two thirds of) the neighbors of an anchor would have the same effect as compromising the anchor. VeIL is capable of gracefully resisting this type of attacks as explained below.

There exist two attack scenarios from a malicious anchor: (a) persisting the same falsified reference location, or (b) spoofing as many false locations as possible (sybil attacks). The former applies to the network with static anchors only, while the latter applies to the network of mobile anchors. Under the first attack scenario, VeIL can still derive sensors' locations such that they best describe the distance measurements in the presence of fake reference location(s), thanks to its cooperative mechanism. Therefore, VeIL resists attacks from compromised anchors by maintaining the network-wide connectivity as well as the local distance relationships. Moreover, VeIL inherently does not require any mobile

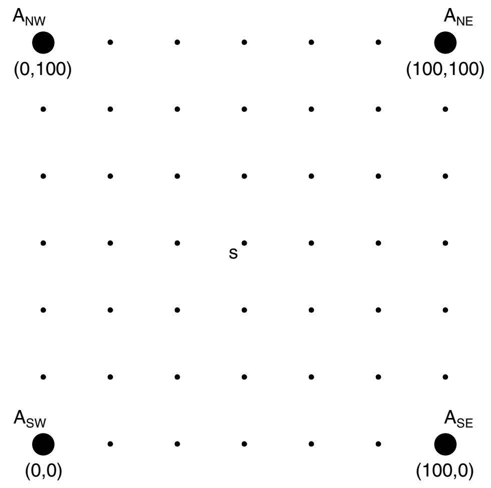


Fig. 4. The simulation environment consisting of 4 anchors and 45 sensors.

anchor, and hence, disallows the use of mobile devices for security reasons, violation of which can be easily checked as follows. Each VeIL-enabled sensor builds a static neighbor list before starting localization (as in Section 3.1), and then binds the reference location with an anchor ID, if it is a neighbor, via the anchor authentication (described in Section 3.2). Hence, any reference location that differs from the initial, authenticated value will be rejected. As a result, the second attack scenario cannot happen in VeIL. By contrast, those protocols relying on mobile anchors become defenseless once the anchor has been compromised or successfully spoofed. Note that VeIL can also be configured to allow mobile anchors although the benefit of using them is marginal in VeIL. In this case, mobile anchors must have higher-level protection than immobile sensors; this may be acceptable since mobile devices (e.g., iPaqs or laptops) are usually equipped with more and better resources (e.g., faster CPUs and more powerful antennae) than static sensors (e.g., motes).

5. PERFORMANCE EVALUATION

We evaluate the performance of VeIL using simulation. We will first describe our simulation environment and the metrics used. Then, we present our simulation results that consist of two parts: (a) quantification of the prediction error of profile management, and (b) evaluation of the attack/anomaly detection capability to mask the effects of malicious neighbors. Please note a comparison with other statistical methods (e.g., Li et al. [2005]; Liu et al. [2005]) is not needed due mainly to the difference in the way the location information is exploited. While VeIL utilizes any available location information (i.e., from both anchors and peer sensors), others solely rely on reference locations from the anchors.

5.1 The Simulation Environment

As shown in Figure 4, our simulation environment consists of a network of 49 sensor nodes deployed on a uniform, 7×7 grid covering an area of 100×100 [m^2].

This simulation setup suffices to demonstrate the effectiveness of VeIL, since it is not the network size but the ratio of the number of compromised to that of normal sensors that determines the localization performance. Accordingly, our simulation focuses on quantification of VeIL's attack detection capability under this simulation scenario. We expect similar results (in terms of attack resolvability) even when different scenarios and/or different network sizes are used.

In Figure 4, the four corner nodes are location-information-equipped anchors, transmitting their (fixed) geographic coordinates. The rest of the network nodes are normal sensors whose locations are unknown, and hence, must be determined. Sensors guess their initial locations completely randomly. Let s be the sensor at the grid center for convenience.

We use the RSS-based ranging technique because it has been widely used by real sensor platforms like motes [Crossbow 2005]. The radio model implemented for our simulation is the one in Patwari et al. [2003] based on real measurements that models the RSS as a log-normal-distributed random variable with its mean power decaying according to the pass loss model. This model captures the characteristics of RSS-based ranging that incurs high estimation errors if two communicating parties are farther away from each other. To deal with these errors, we average 20 RSS measurements before deriving a distance estimate.

The maximum communication range is set to 40 [m] for both anchors and sensors, that is, s accepts i as a neighbor if the distance estimate derived from RSS is smaller than 40 [m]. This effectively limits the number of neighbors for each anchor to be around 7 sensors, and hence, about a half of sensors cannot directly hear from any of the anchors, while the rest can directly hear from at least one anchor. Sensors determine their own list of neighbors based on RSS measurements before starting the localization process.

During the localization process, each sensor executes VeIL (Figure 3) that verifies, computes, and exchanges incremental location updates with its direct neighbors. Throughout the simulation, VeIL is configured with $\lambda = 0.95$ and $\rho = 0.1$ (typical parameter values). The iterative method of Costa et al. [2006] is used to update the location estimates at each iteration, but other iterative schemes can be used as well. The choice of location-update method determines how fast it converges and how computationally efficient it is, without affecting the detection capability of VeIL.

5.2 Metrics for Evaluation

We define and use the normalized prediction error (NPE) of s at iteration k as $NPE_s(k) = \frac{\sum_{i=1}^{n_s} |\alpha_{s,i}(k)|}{c_{s,0}(k)}$, $k \geq M+1$. That is, $NPE_s(k)$ is the sum of absolute values of $\alpha_{s,i}(k)$'s, normalized to $c_{s,0}(k)$. It follows from Equation (9) that $\alpha_{s,i}(M+1) = c_{s,i}(M+1)$, $\forall i$, because $\hat{\mathbf{h}}_{s,i}(M) = \mathbf{0}$, and hence, $NPE_s(M+1)$ always equals 1. We also introduce an individual prediction error (IPE) between s and i at iteration k to quantify the attack-detection capability. Based on Equation (14), we define $ipe_{s,i}(k) = \frac{|\alpha_{s,i}(k)|}{\max\{c_{s,i}(k), c_{\min}\}}$, $k \geq M+1$.

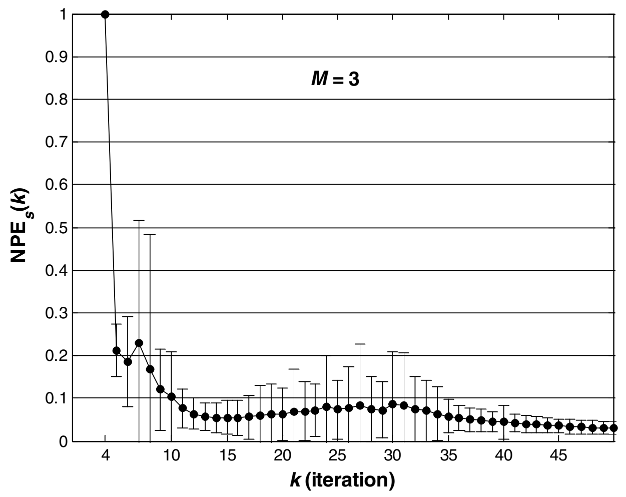


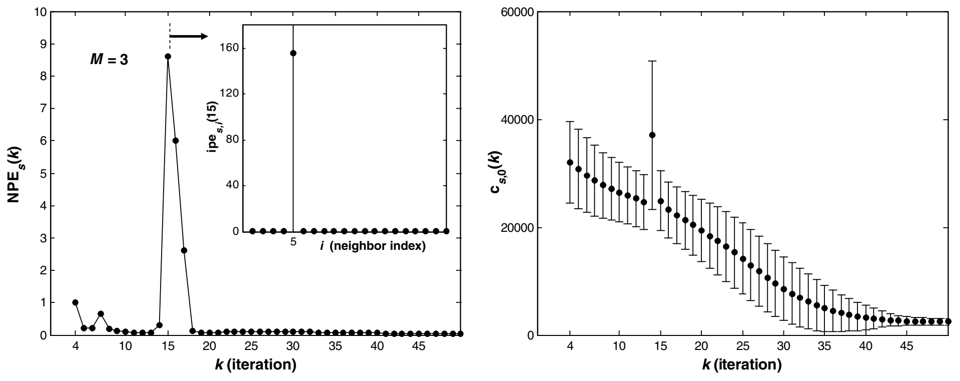
Fig. 5. Attack-free normalized prediction error.

5.3 Performance of the Profile Manager

We evaluate the performance of the profile manager in terms of its prediction accuracy by executing VeIL in an attack-free environment. To quantify this, we collected NPE values of s from 200 independent simulation runs of the localization process. Figure 5 plots the average $NPE_s(k)$, as well as the $(-\sigma_k, +\sigma_k)$ interval, as a function of k, where $M = 3$ and σ_k is the standard deviation of NPE measurements at iteration k. The results for $M = 4$ and 5 were similar to this. From this figure, we make the following observations. First, NPEs were mostly less than 0.1, demonstrating high accuracy of the profile manager. Second, NPE was around 0.2 at the $(M+2)^{\text{nd}}$ iteration, meaning that it constructed a ready-to-use profile pretty quickly, that is, right after the filter gets activated. Third, the profile manager incurred small processing and storage overheads thanks to the small order (=3) of the filters. As mentioned in Section 3.5.3, the processing overhead of s for updating the profile is $\mathcal{O}(M^2)$ per neighbor, which is acceptable even for resource-limited sensors when $M = 3$. In summary, our proposed profile manager based on adaptive filtering captures the localization behavior in as compact a form as possible, thus achieving both accuracy and computational efficiency. Finally, NPE slowly increased with higher standard deviation during iterations 12 to around 25 due mainly to the fact that some of the sensors were not stabilized yet during this earlier stage of localization.

5.4 Performance of the Attack Detector

To evaluate the attack detector's performance, we simulate VeIL under the attack scenarios of (a) a single attack source and (b) multiple simultaneous attack sources. Described below are the simulation results and their analyses for each of the two scenarios.



(a) Normalized and individual prediction errors (b) Local cost

Fig. 6. Attack-detection capability: a single false location announcement at iteration 14.

5.4.1 *Defense against a Single Attack Source.* In this simulation, a malicious neighbor (say, sensor 5) of s injects at iteration 14 false location information that deviates from the expected location by 40 [m], where the direction of deviation is determined randomly. This attack scenario is suitable for evaluating the VeIL's ability to handle false location announcements. We carried out 200 simulation runs under this attack scenario, and measured/computed $NPE_s(k)$, $ipe_{s,i}(k)$, and $c_{s,0}(k)$.

Figure 6(a) plots both $NPE_s(k)$ and $ipe_{s,i}(k)$. The former quantifies the attack strength, while the latter identifies the source of the false information. We observe that the attack occurred at iteration 14 increased $\alpha_{s,5}(14)$ in proportion to the attack strength, and boosted $\alpha_{s,5}(15)$ by orders of magnitude. Moreover, the next two iterations also exhibited unusually large prediction errors. This implies that the adversarial cost disrupted the prediction mechanism while it resided in the profile manager. Clearly, these results agree with our analysis in Section 3.6. In Figure 6(b), we also plot $c_{s,0}(k)$ as a function of k. The figure shows the local cost created a spike at iteration 14, thus causing the test of Equation (15) to fail. This is an evidence that the increase in $\alpha_{s,5}(14)$ is due to an attack.

Based on these results, we constructed multiple layers of defense mechanisms against attacks as follows. First, the tests based on Equations (14) and (15) serve as the first line of defense that diagnoses and combats attacks as early as possible. Second, VeIL checks and monitors the strength of $\alpha_{s,5}(\cdot)$ for the next M iterations to uncover the attacks that somehow evaded the first line of defense.

5.4.2 *Defense against Multiple Attack Sources.* We now evaluate the VeIL's capability to detect multiple malicious nodes that have simultaneously announced false locations. Figure 7 presents the IPE of s at iteration 15 when four malicious neighbors (whose indices are 5, 10, 13, and 18) simultaneously mounted attacks of varying strengths that deviate from the desired values by 20, 50, and 80 [m]. We make the following observations from the figure. First, $ipe_{s,i}(15)$ increased very rapidly with the attack strength, demonstrating VeIL's

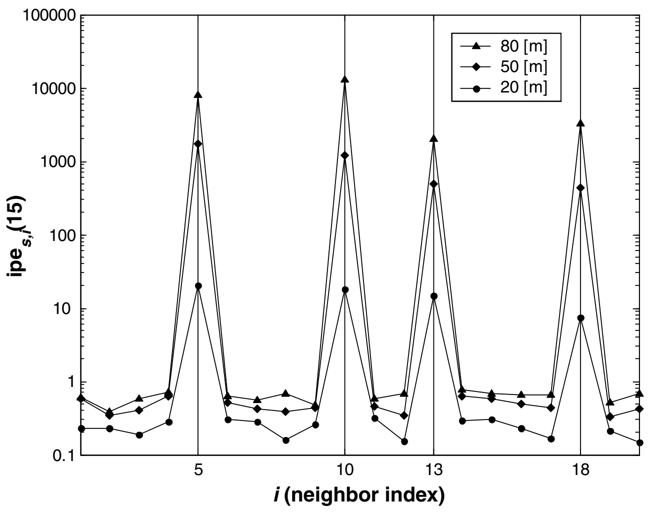


Fig. 7. Individual prediction errors.

effectiveness in countering location-targeted attacks. Second, VeIL accurately detected attacks incurring small perturbations (e.g., a half of the communication range). This demonstrates the VeIL's very high resolution in detecting attacks. Finally, VeIL preserved its attack-detection capability regardless of the number of attack sources as far as they are less than one third of the total number of neighbors, which is obvious from the fact that VeIL separately maintains an adaptive filter for each neighbor.

6. CONCLUSION

In this article, we proposed a novel attack-tolerant localization scheme, called VeIL, for a large-scale sensor network deployed with only a small number of less-capable anchors. The use of spatio-temporal correlation among adjacent nodes played a key role in developing VeIL as a cooperative intrusion/anomaly detection system tailored to localization that consists of (a) adaptive management of the profile for normal localization behavior, and (b) distributed detection of false locations via comparison with the thus-managed profile. We then performed in-depth analysis and evaluation of VeIL's security properties and capabilities, and demonstrated the high-level attack-tolerance and the feasibility of VeIL on resource-limited sensors, in that VeIL successfully defeats many critical attacks while incurring only small overheads.

As a future work, we will investigate a weighted version of MDS algorithm as the underlying localization algorithm to develop a systematic way of detecting and blacklisting malicious sensors. That is, instead of the blacklisting scheme that counts how many times a sensor has been suspected of cheating, we will devise a soft threshold scheme in which the sensors' weights are adaptively adjusted in proportion to their levels of trustworthiness.

REFERENCES

BASALAJ, W. 2001. Proximity visualization of abstract data. Tech. rep.

BORRIELLO, G., LIU, A., OFFER, T., PALISTRANT, C., AND SHARP, R. 2005. Wireless acoustic location with room-level resolution using ultrasound. In *Proceedings of the 3rd Annual Conference on Mobile Systems, Applications and Services (MobiSys'05)*. ACM, New York.

BULUSU, N., HEIDEMANN, J., AND ESTRIN, D. 2000. GPS-less low cost outdoor localization for very small devices. *IEEE Person. Comm. Mag. 7*, 5.

CAPKUN, S. AND HUBAUX, J.-P. 2004. Secure positioning in sensor networks. Tech. rep. EPFL/IC/200444.

CHAN, H., PERRIG, A., AND SONG, D. 2003. Random key predistribution schemes for sensor networks. In *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, Los Alamitos, CA.

COSTA, J., PATWARI, N., AND HERO III, A. O. 2005. Achieving high-accuracy distributed localization in sensor networks. In *Proceedings of International Conference on Acoustics, Speech, and Signal Precessing (ICASSP'05)*. IEEE, Los Alamitos, CA.

COSTA, J., PATWARI, N., AND HERO III, A. O. 2006. Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *ACM Trans. Sens. Netwo. 2*, 1.

CROSSBOW TECHNOLOGY. 2005. MICA, MICA2 motes & sensors. Tech. rep.

DOUCEUR, J. 2002. The sybil attack. In *Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*. Springer, Berlin.

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 (CCS'02)*. ACM, New York.

HAYKIN, S. 1991. *Adaptive Filter Theory*, 2 Ed. Prentice-Hall, Upper Saddle River, NJ.

HE, T., HUANG, C., BLUM, B. M., STANKOVIC, J. A., AND ABDELZAHER, T. 2003. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Commuting and Networking (MobiCom'03)*. ACM, New York.

HU, L. AND EVANS, D. 2004. Localization for mobile sensor networks. In *Proceedings of the 10th International Conference on Mobile Commuting and Networking (MobiCom'04)*. ACM, New York.

JAIN, R., PURI, A., AND SENGUPTA, R. 2001. Geographical routing using partial information for wireless ad hoc networks. *IEEE Person. Comm. 8*, 1, 48–57.

JI, X. AND ZHA, H. 2004. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*. IEEE, Los Alamitos, CA.

LAZOS, L. AND POOVENDRAN, R. 2004. SeRLoc: secure range-independent localization for wireless sensor networks. In *Proceedings of the 3rd ACM Workshop Wireless Security (WiSe'04)*. ACM, New York.

LI, Z., TRAPPE, W., ZHANG, Y., AND NATH, B. 2005. Robust statistical methods for securing wireless localization in sensor networks. In *Proceedings of 4th International Symposium on Information Processing in Sensor Networks (IPSN'05)*. ACM, New York.

LIU, D., NING, P., AND DU, W. 2005. Attack-resistant location estimation in sensor networks. In *Proceedings of 4th International Symposium on Information Processing in Sensor Networks (IPSN'05)*. ACM, New York.

MISHRA, A., NADKARNI, K., AND PATCHA, A. 2004. Intrusion detection in wireless ad hoc networks. *IEEE Wire. Comm. 11*, 48–60.

NICOLESCU, D. AND NATH, B. 2001. Ad-hoc positioning systems (APS). In *Proceedings of the Global Telecommunications Conference (GLOBECOM '01)*. IEEE, Los Alamitos, CA.

PARK, T. AND SHIN, K. G. 2005. Soft tamper-proofing via program integrity verification in wireless sensor networks. *IEEE Trans. Mobile Comput. 4*, 3.

PATHIRANA, P. N., BULUSU, N., SAVKIN, A. V., AND JHA, S. 2005. Node localization using mobile robots in delay-tolerant sensor networks. *IEEE Trans. Mobile Comput. 4*, 3.

PATWARI, N., HERO III, A. O., PERKINS, M., CORREAL, N. S., AND O'DEA, R. J. 2003. Relative location estimation in wireless sensor networks. *IEEE Trans. Signal Process. 51*, 8.

PERRIG, A., SZEWCZYK, R., WEN, V., CULLER, D., AND TYGAR, J. D. 2001. SPINS: security protocol for sensor networks. In *Proceedings of the 7th Annual International Conference on Mobile Commuting and Networking (MobiCom'01)*. ACM, New York.

PRIYANTHA, N. B., BALAKRISHNAN, H., DEMAINE, E. D., AND TELLER, S. 2005. Mobile-assisted localization in wireless sensor networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*. IEEE, Los Alamitos, CA.

SASTRY, N., SHANKAR, U., AND WAGNER, D. 2003. Secure verification of location claims. In *Proceedings of the 2nd ACM Workshop Wireless Security (WiSe'03)*. ACM, New York.

SAVARESE, C., RABAY, J., AND LANGENDOEN, K. 2002. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX Annual Technical Conference*. USENIX, Berkeley, CA.

SHANG, Y., RUML, W., ZHANG, Y., AND FROMHERZ, M. P. J. 2003. Localization for mere connectivity. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'03)*. ACM, New York.

WATERS, B. R. AND FELTEN, E. W. 2003. Secure, private proofs of location. Tech. rep. Princeton University, TR-667-03.

WHITEHOUSE, K. AND CULLER, D. 2002. Calibration as parameter estimation in sensor networks. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Application*. ACM, New York.

YOUSSEF, M. AND AGRAWALA, A. 2005. The horus WLAN location determination system. In *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys'05)*. ACM, New York.

ZHANG, Y. AND LEE, W. 2000. Intrusion detection in wireless ad hoc networks. In *Proceedings of the 6th Annual International Conference on Mobile Commuting and Networking (MobiCom'00)*. ACM, New York.

Received February 2006; revised January 2007; accepted October 2007