Secure Localization and Location Verification in Wireless Sensor Networks

Yingpei Zeng† Jiannong Cao‡
†State Key Laboratory for Novel Software Technology
Nanjing University, Nanjing, P.R. China
Email: {zyp,jhong}@dislab.nju.edu.cn

Jue Hong† Li Xie†
‡Department of Computing
Hong Kong Polytechnic University, Hong Kong
Email: csjcao@comp.polyu.edu.hk, xieli@nju.edu.cn

Abstract

Sensors’ locations are important to many wireless sensor networks (WSNs). When WSNs are deployed in hostile environments (e.g., battlefield), two problems about sensors’ locations need to be considered. First, the attackers may attack the localization process to make the estimated locations incorrect. Second, since sensor nodes can be compromised, the base station may not trust the locations reported by sensor nodes. Researchers have proposed two techniques, secure localization and location verification, to solve the two problems respectively. In this paper we survey the state of research in both secure localization and location verification.

1. Introduction

Wireless sensor networks (WSNs) are composed of small, low cost, and low power sensor nodes [1]. Many applications have been proposed for WSNs. They range from environmental applications like volcano monitoring to military applications like battlefield surveillance [1]. In many applications WSNs are deployed in unattended and even hostile environments, where we must consider the security issues to ensure the operation of the WSNs.

Many WSNs require the knowledge of sensors’ locations. First, the data collected by sensors usually should be bound with locations, e.g., a truck is detected at location loc. Second, many network operations also depend on the locations of sensors, e.g., geographic routing [2], geographic key distribution [3], and location-based authentication [4]. Now many localization algorithms have been proposed in the literature.

When WSNs are deployed in hostile environments, the attackers may attack the localization process to make the estimated locations incorrect. Incorrect locations may lead to severe consequences, e.g., wrong military decisions on the battlefield and falsely granting access rights to people. Thus it is important to ensure the correctness of sensors’ locations.

For ensuring the correctness of sensors’ locations, we should consider the need of sensors and the need of others using sensors’ locations (mainly the base station). At the sensor side, as we mentioned, sensors themselves need to get their correct locations (e.g., to tag the sensed data), so we need secure location determination, which we call secure localization in the paper. At the base station side, the base station (BS) also needs to ensure the sensors’ locations it gets are correct (e.g., to make sure the event really happened there). This is because when the BS needs to learn sensors’ locations from sensors (i.e., is node-centric localization as we will explain later), the sensor nodes may be compromised and intentionally report false locations. Thus we need to verify the location claims. We call this as location verification.

In this paper we first describe the secure localization problem and the location verification problem (Section 2), and review the known attacks in them (Section 3). Then we describe and classify the state of research in both secure localization (Section 4) and location verification (Section 5). Finally we present the conclusion and several open research problems (Section 6). Different from existing review articles [5], [6], we survey the two related fields, secure localization and location verification, at the same time to provide a more comprehensive review.

2. Problem Statement

In the section we define the problems that secure localization and location verification try to solve. Note that before introducing secure localization, we describe the general localization process first.

2.1. Localization

Usually the sensor network contains two kinds of nodes: common nodes and beacon nodes. Common nodes do not know their locations, and beacon nodes know their locations (e.g., by GPS). Then, the localization process is to estimate the locations of the common nodes. Usually, the localization process can be divided into two steps (with an optional refinement step), as shown in Figure 1:

- Information collection: The information for localization is collected, which may include the connectivity, distances, and angles, as well as the locations of beacons.
- The distances between nodes in single hop can be
measured by received signal strength indicator (RSSI), time of arrival (ToA), or time difference of arrival (TDoA) [7]; the distances between nodes multihop-away can be measured by DV-hop [8] or DV-distance methods [8]. The angles can be measured by angle of arrival (AoA) [9].

- **Location computation**: The locations are computed with the collected information. Many algorithms have been proposed. Simple algorithms include trilateration [8], multilateration [7], and triangulation [9]. Also, more complicated algorithms have been proposed, e.g., MDS-MAP for localizing the network as a whole [10], RobustQuad for coping with noisy measurements [11], and Sweep [12] for localization in sparse networks.

The optional refinement step is for iteratively computing locations with newly calculated locations (e.g., the localized node will become new beacons [7]) or with new computation methods (e.g., in [13]–[15], new methods will be executed after obtaining nodes’ coarse locations).

The localization systems can be classified into range-based and range-free. In range-based systems, the distances or angles between nodes need to be measured in the information-collection step. Range-free systems do not have such requirements. Thus, the range-free systems usually do not require any additional hardware.

The localization systems can also be classified into node-centric and infrastructure-centric [16]. In the former systems sensor nodes compute their locations by themselves. In the latter systems the infrastructure (we refer to the infrastructure as the BS and any other nodes the BS trusted, e.g., special mobile stations) computes the locations of nodes.

### 2.2. Secure Localization

Secure localization is to make the above localization process still correct under attacks. It may require additional hardware to defeat attacks. The classification of secure localization systems may also follow the classification of general localization systems in the above subsection. We next briefly describe the adversary model in secure localization.

**Adversary model**: The goal of the adversary is to make the nodes (i.e., in node-centric localization) or the infrastructure (i.e., in infrastructure-centric localization) obtain false estimated locations. He can compromise partial nodes (common nodes and beacons). He can intercept, jam, and replay signals in any transmission medium.

### 2.3. Location Verification

When the infrastructure is managing the network based on sensors’ reported locations, e.g., processing the data binding with locations or authenticating sensors based on their locations, it may not trust these reported locations.

We consider the cases in two types of localization systems. If the localization system is infrastructure-centric, the infrastructure will trust the estimation locations, because the locations are computed by itself (the locations may also be incorrect, but securing the localization is the only thing it can do). However, if the localization system is node-centric, the infrastructure will not simply trust the estimation locations. This is because even the locations are obtained through secure localization, the nodes may be compromised and intentionally report false locations. Adding tamper-resistant hardware for honestly reporting locations is an approach; however it will increase the cost of node and is proved to be problematic in practice [18].

Thus, when localization system is node-centric, location verification is needed to verify the claimed locations of sensors. In location verification systems, the sensor node to be verified is called the prover and the infrastructure is called the verifier. We note that in some scenarios verifying that the sensor node is inside a given region (but not precisely at a position) is sufficient, e.g., verifying that a node is inside a coffee shop for judging the qualification for some services. We next briefly describe the adversary model in location verification.

**Adversary model**: The goal of the adversary is to make the verification failed, i.e., correct location claims from normal provers are verified as incorrect and are rejected, but false location claims from compromised provers are verified as correct and are accepted. Similar to secure localization systems, the adversary can compromise partial nodes (common nodes and beacons). He can also intercept, jam, and replay signals in any transmission medium.

### 3. Known Attacks

Many attacks can be launched in localization systems and location verification systems.

1. **Sensors using other sensors’ locations may also not trust other sensors’ claimed locations, however they usually trust the infrastructure, so it is not a problem when sensors’ locations are verified by the infrastructure. Also, some researchers consider such location verification scenario: sensors do not trust their locations computed by themselves, so they verify their locations before using them, e.g., in [17]. However we think in such scenario we can use secure localization instead, and so we only consider location verification by the infrastructure.**
Range-change Attack: In this attack the attacker may decrease or increase the range measurements between any nodes. In single-hop case, if the measurement is RSSI-based, the attacker can decrease or increase the transmission power of the senders when the senders are compromised (when the sender is a normal node, the attacker can jam its signal and replay it with lower or higher transmission power). If the measurement is ToA- and TDoA-based, the attacker can delay the transmission of packets. In multihop case, to distort the range measurements, the attacker can decrease or increase the hop counts in DV-hop based systems [8], and decrease or increase the distance in each single hop in DV-distance based systems [8]. Note that this attack has effects on both localization systems and location verification systems. For example, reducing the range measurement between node \( A \) and \( B \) may distort the estimated location of \( B \) if \( A \) is a beacon, and may also make \( A \) wrongly believe that \( B \) is within a given region if \( A \) is a verifier.

Impersonation: In this attack the attacker impersonates other nodes in the network. For example, in localization systems, the attacker may impersonate beacon nodes to broadcast false locations. In location verification systems, the attacker may impersonate a victim prover to make the verifier believe the prover is at the attacker’s location. This attack can be defeated by authentication.

Wormhole attack: In this attack the attacker records packets at one location in the network, tunnels them to another location, and replays them [19]. The attacker may directly launch the attack (i.e., receiving and replaying packets with private radios and tunneling with a private channel), or launch with two compromised nodes (i.e., one for receiving and another for replaying and the tunneling is finished by routing in the WSN). The replay attack, which is to maliciously forward heard packets (e.g., forward packets heard from beacons [20]), can be regarded as a zero-tunnel-length wormhole attack. In localization systems, wormhole attack will make the beacons in one side appear at another side and make the information collected for localization erroneous. In location verification systems, the attacker may tunnel the packets of a victim prover to another location and make the verifier believe that the prover is at the false location.

Sybil attack: In this attack the attacker has obtained several node identities, and then he can make one compromised node masquerade as several nodes at the same time. For example, in localization systems, one compromised node may masquerade as several beacons (their identities are compromised by the attacker), and send false information.

Location-reference attack: This attack is launched in localization systems (e.g., [7], [8], [13], [21], [22]) in which each common node gets a location-reference set \( \{< \text{loc}_i, d_i >\} \) for localization (\( \text{loc}_i \) is the location of beacon \( i \) and \( d_i \) is the distance between the beacon and the common node). In this attack the attacker may make the compromised beacons broadcast false locations, and/or may distort the distance measurements between beacons and common nodes (i.e., may contain range-change attacks). In a word, the attacker may change partial location references in the whole location-reference set. According to the smart level, the attacks can be classified into three types: uncoordinated attacks, collusion attacks, and pollution attacks. Exemplary scenarios are shown in Figure 2. In uncoordinated attacks, different bad location references are to mislead the common node to different false locations, e.g., \( P1 \) and \( P2 \) in the figure. In collusion attacks, all the bad location references are to mislead the common node to random but the same false location. This attack is more powerful, however it is still can be defeated when normal location references are in the majority [22]. In pollution attacks, all the bad location references are to mislead the common node to a specially chosen false location, which also conforms to partial normal location references. This attack is the most powerful one and in some cases it may succeed even when normal location references are in the majority [23].

4. Solutions for Secure Localization

Many secure localization systems have been proposed. As we mentioned they can be classified into two types, node-centric and infrastructure-centric, based on the place where sensors’ locations are computed. Also, in infrastructure-centric secure localization systems, the obtained locations need not be verified by the infrastructure.

Based on their goals, existing solutions can be classified into three methods: 1) Prevent the adversary from producing erroneous information (the prevention method), 2) Detect and revoke the nodes that producing erroneous information (the detection method), and 3) Filter the received erroneous information in location computation (the filtering method).
4.1. Node-centric Secure Localization

The prevention method: Researchers proposed several solutions following the prevention method [24]–[28], i.e., prevent nodes from generating erroneous information to the localization system. In SeRLoc [24], Lazos et al. proposed to employ special trusted nodes called locators to replace beacons. The locators are equipped with sectored antennas and have longer transmission range. When a node hears multiple locators, it computes the center of gravity of the sectors corresponding to locators as its location. The same authors latter proposed an improved method HiRLoc [25], which can achieve higher accuracy through rotatable antennas and variable transmission power.

In [26], Capkun et al. proposed SPINE based on the verifiable multilateration (VM) technique introduced in the same paper. In VM, if a node is inside the triangle formed by three nodes with known locations, through distance bounding [29], its location can be uniquely determined. In SPINE, all the distance measurements are verified by triangles around them formed by sensor nodes. Thus nodes cannot produce erroneous distance measurements.

In [28], combining the techniques in SeRLoc [24] and VM [26], Capkun proposed ROPE. In ROPE each node obtains its exact location by VM when it is inside at least one triangle formed by locators, and still estimates its location by center of gravity when it is not inside any triangle. In [30], Zeng et al. proposed SHOLOC to prevent the compromised nodes from reducing the hop counts in hop-count based localization algorithm. Their method is to represent the value of hop count by the number of hash operations on a nonce, thus compromised nodes cannot reduce the hop counts.

The detection method: Two solutions have been proposed in this category [20], [31], [32], and they both focus on detecting malicious beacons, because beacons have great impact on localization. In [20], Liu et al. proposed to use detecting beacons to detect malicious beacons broadcasting false locations. The detecting beacons will send requests to beacons to be checked same as common nodes. When they receive the replied locations from the beacons being checked and measure the distances between them, they will compare the measured distances with the distances computed by using their locations and the replied locations. If the distances are inconsistent, the beacons being checked are malicious and will be revoked. The authors also proposed a method based on round trip time to filter the replayed beacon signals and avoid false positives.

In DRBTS [31], [32], Srinivasan et al. generalized the solution by Liu et al. [20] by employing beacons to maintain reputations for their neighbor beacons. Each beacon computes reputations of its neighbor beacons based on the overheard location reply as well as the reputation value heard from other beacons. Common sensor nodes will only use beacons trusted by other beacons to compute its location.

The filtering method: Many works have been done for filtering the impact of received erroneous information [21], [22], [33]–[37]. They all focus on filtering the bad location references in location-reference set since in many algorithms nodes compute locations based on location-reference sets [7], [8], [13]. In [21], [33] Liu et al. proposed three different ARMMSE algorithms and a voting-based algorithm. The basic idea of ARMMSE is to obtain a subset of location references, which satisfies that the mean square error of the location computed by the subset is below a threshold \( \tau \). The three different ARMMSE algorithms are to obtain such subset in different ways. In the voting-based algorithm, they first divide the minimum rectangle that covers all the location references into cells. Then each location reference votes to the cells which conform to its observation. Finally the algorithm selects the centroid of the cell(s) with the highest vote as estimated location, or further divides the selected cell(s) and starts the vote again to improve the precision.

In [22] Li et al. proposed to use LMS [38] to filter the bad location references. Different from traditional methods that minimize the mean square error, LMS method is to minimize the median of square errors: \( \text{loc}_0 = \arg \min_{\text{loc}_0} \text{med}_i \left[ \text{dist}(\text{loc}_0, \text{loc}_i) - d_i \right]^2 \), where \(< \text{loc}_i, d_i > \) is the \( i \) location reference, \( \text{loc}_0 \) is the estimated location, and \( \text{dist} \) is to compute the Euclidean distance between two locations. Experiments show that LMS can filter the impact of bad location references.

In [34], [35], Misra et al. proposed a method to filter compromised beacons when distance bounding [29] is used and the attackers can only enlarge the distances. Their method is to compute the geometric center of the intersection of circles corresponding to location references.

In [37] Zhong et al. proved that when there are no more than \( \frac{n-3}{2} \) compromised beacons (\( k \leq \frac{n-3}{2} \)) \(^4\), we can definitely compute the location of node with an error bound proportional to \( \epsilon \), where \( n \) is the number of beacons, \( k \) is the number of normal beacons, and \( \epsilon \) is the measurement error. However, such result is proved under the condition that \( \epsilon \) is ideally small. In [23] Zeng et al. showed that the attacker can still seriously distort the estimated location when \( k \leq \frac{n-3}{2} \) holds and \( \epsilon \) is practically small. In [37] Zhong et al. also proposed two algorithms to compute the location, based on finding an arc that is inside \( k+3 \) rings (each ring corresponding to a location reference) and finding an intersection point that is inside \( k+3 \) rings respectively, because the localization errors are bounded by the error bound if the estimated location is inside \( k+3 \) rings.

\(^4\) It is equal to say the condition \( g \geq k+3 \) should hold, where \( g \) is the number of compromised beacons. In [34] a similar result is proved.
4.2. Infrastructure-centric Secure Localization

Infrastructure-centric localization systems usually follow the prevention method, because they usually employ reliable infrastructure, and do not have vulnerable special nodes like beacons. Capkun et al. [16], [39] proposed a method to localize nodes based on covert base stations (CBS). These CBS are hidden from the nodes and attackers. First, the public base station (PBS) sends a nonce. When a node replies to the nonce, all the CBS will compute its location together based on the TDoA method. Then if the sum of the actual time differences deviates from the supposed values over a threshold, an attack is detected and the estimated location is rejected, otherwise the location is accepted.

Zhang et al. [40] proposed SLS for UWB sensor networks. The authors assume that there is a set of trusted anchors which can perform group movement in the deployment field. In SLS, first, each anchor performs an algorithm called K-Distance to measure the distance between the anchor and the node to be localized. K-Distance is to use the median of $K$ rounds of ToA to compute the distance. K-Distance can prevent the attackers from shortening the measured distance; it is similar to the distance bounding technique, however the prover here is honest and the processing times of the signals are known. Second, anchors send the measured distances to the anchor leader to compute the location of the node. Third, SLS employs a location validity test by checking whether the location is inside the polygon formed by all the anchors. This test process is similar to VM [26], [27], but here the number of vertices of the polygon can be more than three.

Anjum et al. [41] proposed SLA to securely localize nodes based on transmission range (TR) variation at the anchor nodes. The anchors are assumed to be reliable and can vary their TR to several values. In the localization the BS let anchors transmit different nonces at different TRs. Each sensor then sends its received nonces to the BS. The BS computes sensors’ locations based on the unique set of nonces at any location.

4.3. Comparison of Secure Localization Solutions

In secure localization systems, node-centric solutions are more popular than infrastructure-centric solutions. This can be explained from the cost consideration. Usually in infrastructure-centric secure localization systems, reliable infrastructure is needed. Traditional beacon nodes may be compromised, so we should deploy extra reliable hardware, with higher cost. However, in some circumstances demanding higher security, the higher cost may be justified since in infrastructure-centric secure localization systems, there is no need for location verification.

We list the classification of existing solutions in Table 1. Comparing with solutions in other two methods, the solutions in the filtering method usually do not need to deploy any addition hardware. They just need to add authentication support to the existing protocols, and then replace the vulnerable location-estimation methods such as MMSE with new attack-resistant methods. Some solutions following other methods also do not need additional hardware [30]–[32].

The three methods are from radical to conservative, and they may operate in the defend-in-depth manner. The first method tries to make the attackers unable to produce erroneous information; in some cases it may be too costly or impossible. For example, the beacon nodes usually are vulnerable and compromised beacons may broadcast false locations. Then we may need the second method which is to detect such sources of errors and revoke them. Finally, if partial erroneous information escapes from the detection, we need the final line of defense, which is to filter the erroneous information (i.e., the third method). In fact, some schemes already adopt more than one method in the design [25], [30].

Table 1. Secure localization systems comparison.

<table>
<thead>
<tr>
<th>Node-centric</th>
<th>Prevention method</th>
<th>Detection method</th>
<th>Filtering method</th>
<th>Additional hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>SeRLoc [24], HIRLoc [25], SPINE [26], [27], ROPE [28], SHOLOC [30]</td>
<td>Liu et al. [20], DRBTS [31], [32]</td>
<td>HiRLoc [25], SHOLOC [30], ARMMSE [21], [33], LMS [22], ROSETTA [34], [35], Kiyavash et al. [36], Zhong et al. [37]</td>
<td>CBS [16], [39], SLS [40], SLA [41]</td>
<td></td>
</tr>
</tbody>
</table>

5. Solutions for Location Verification

Based on the goals of verification, we classify the existing location-verification solutions into two types: in-region [4], [28], [29], [42], [43] and single-position [16], [17], [39], [44]–[48]. The former is to verify that whether nodes (provers) are inside a given region. The latter is to verify that whether nodes (provers) are at given positions.

5.1. In-region Verification

Several solutions are proposed based on the distance-bounding technique. Brands and Chaum first proposed distance bounding in [29] to make the prover unable to reduce its distance to the verifier (for defeating the mafia fraud). First, the prover (P) sends a commitment on a bit string
to the verifier (V) (e.g., send the hashed value, by
a collision-free hash function, of the bit string), and V
prepares a random bit string $\alpha_i$. Second, the low-level
distance-bounding exchanges start: V sends bit $\alpha_i$ to P,
and P sends bit $\beta_i = \alpha_i \oplus m_i$ to V immediately after he
receives $\alpha_i$. Third, P opens the commitment and sends the
signature $\text{sign}(\alpha \parallel \beta)$ to V, and V computes an upper-bound
on its distance to P based on the maximum of delay times
between sending out a bit $\alpha_i$ and receiving bit $\beta_i$ back. Such
distance bounding using RF (radio frequency) signal requires
dedicated hardware [26] (because we need to measure time
with nanosecond precision).

In [4] Saistry et al. proposed the Echo protocol (similar
to distance bounding) to verify that whether the prover is
inside a given region. The region is covered by small circular
regions and each verifier is in charge of the verification
in a small region. In the Echo protocol, first, the prover
P broadcasts its location $l$. Second, the verifier V sends a
nonce to P using RF and starts the timer, and the prover
P immediately echoes the nonce back using ultrasound.
Finally, V uses the elapsed time to compute the distance
and judges that whether V is inside its circular region. The
Echo protocol is similar to the distance bounding protocol
[29], however the outgoing and incoming signals of Echo
are (RF, sound) (need no precise clock), and so Echo does
not require sophisticated hardware.

In [43] Vora et al. proposed a new method to achieve the
same goal as [4]. They divided the verifiers into acceptors
and rejectors. The acceptors are deployed inside the pro-
tected region and rejectors are deployed at the boundary of
the region. The verification process is the prover step by step
increases its signal strength and broadcasts a signal, until a
verifier hears the signal and responds. The verifiers accept
the prover if none of the rejectors hears it during the process.

5.2. Single-position Verification

Base on the number of nodes verified at a time, we can
further classify the verification algorithms into two types:
batch-verification [47], [48] and single-node-verification
[16], [17], [39], [44]–[46]. The former is to verify a batch
of nodes at a time, and the latter is to verify nodes one by
one.

Batch-verification: In [48] Wei et al. proposed two al-
gorithms running at a Verification Center (VC) to verify
the locations of nodes: GFM and TI. GFM is to detect the
abnormal sensor locations based on the inconsistency in four
derived matrices. These four matrices represent the observed
neighbors and neighbors computed by estimated locations.
The authors also proposed four metrics computing over the
four matrices for characterizing abnormal sensors. In TI, an
iterative process is run to update the indicator value of each
node. In such process each node observing a node $i$ gives its
indicator value computed from geographical relationship to
judge whether the node $i$ has abnormal location. TI gets the
verification result and stops updating the indicator of a node
if its indicator grows beyond the threshold or converges.

In [47] Hwang et al. proposed an algorithm for each node
to detect the phantom nodes in its neighborhood. Here the
algorithm runs a process for given times. In each run, the
node first creates a local map randomly using two other
neighbors. Then in each such map, we try to find the largest
consistent subset. The finding method is to check each node
that whether the measured ranges are consistent with the
ranges computed using the node’s location in the map. At
last, the largest subset in all the runs is selected, and it
contains all the consistent nodes in the node’s neighborhood.

Single-node-verification: In [17] Du et al. proposed LAD,
which is to use deployment information to detect localization
anomaly. Considering sensors with group-based deploy-
ment, each node can be assumed to follow two-dimensional
Gaussian distribution, which is centered at the deployment
point of that node’s group. Then the authors proposed three
metrics for each node to detect anomaly: the Diff, the Add-
all, and the Probability metrics. Take the Diff metric for
example, it represents the difference between the actual
observation and the expected observation (an observation
is a vector, in which the $i$ value represents the number of
neighbors in $i$ group). The threshold values of the metrics
indicating anomaly are obtained through training. We note
that the LAD is executed by each node itself; however it is
easy to be executed at the BS.

In [16], [39] Capkun et al. also proposed to use covert
base stations (CBS) and mobile base station (MBS) to
verify reported locations of nodes. In the CBS case, the
node to be verified broadcasts a RF signal and a sound
signal. Then CBS can calculate the distance between the
CBS and the node based on TDoA. Since each CBS knows
its location, the calculated distance is compared with the
distance computed using the reported location and CBS’
location. If the difference is beyond a threshold, the reported
location is rejected. In the MBS case, the process is similar.
The MBS first requires the node to broadcast the RF and
sound signals after given time $T_R$. After that time, the MBS
has moved to a different location not known by the node,
so it can check the reported location similarly as a CBS.

In [45], [46] Ekici et al. proposed a probabilistic method
(PLV) to verify a node’s location. Some trusted verifiers
knowing their locations are deployed in the network. When
the verification starts, the node floods its location in the
network, with a hop count field. Then each verifier can
gen the number of hops between the node and verifier and
compute the distance between them. Based on two values,
each verifier computes two probabilities: one represents
the judgment that such value pair do occur, and another
represents the confidence of the judgment. Finally a central node collects the information from all verifiers and gives the final decision on acceptance and rejection.

5.3. Comparison of Location Verification Solutions

We list the classification of existing solutions in Table 2. Some single-position verification algorithms do not need any additional hardware [17], [47], [48]; however, in-region verification algorithms usually need additional hardware to represent the region to be protected or verified.

In single-position verification systems, single-node-verification systems usually are more efficient than batch-verification systems when we want to verify some critical nodes, e.g., the nodes which reported events. However batch-verification systems are more appropriate when we want to verify all the nodes at one time.

6. Conclusion and Open Research Problems

In this paper we described the problems that secure localization and location verification try to solve. We also discussed the known attacks in localization and location verification. Finally we described and classified existing solutions in both secure localization and location verification.

A number of research problems remain in the area of secure localization and location verification. First, no solution for secure localization in multihop & range-based systems exists. Unsecured algorithms like RobustQuad [11] and Sweep [12] do exist, but most of existing secure localization approaches cannot be directly used in such algorithms to protect them (many existing secure localization approaches [21], [22], [33]–[37] can only be applied in simpler algorithms like DV-hop [8]). Collecting information through multipath may be a plausible way.

Second, very few works exist for secure localization in some special WSNs, e.g., sparse WSNs [12] and mobile WSNs [49]. These networks raise many challenges in the algorithm design. Before designing solutions for such WSNs, we would better learn their restrictions from existing insecure solutions.

Third, no solution exists for location verification for one node at a time (i.e., single-node-verification) without any additional infrastructure support (e.g., no verifier and special hidden/mobile stations) and without any deployment information. Such work is challenging; however it will definitely be interesting. Possible solutions may utilize within-n-hop neighbors.

Acknowledgment

This work is supported in part by Hong Kong Research Grant Council under CERG grant PolyU 5102/07E, the Hong Kong Polytechnic University under the ICRG grant G-YF61, Natural Science Foundation of China under Grant No.60873154, and Natural Science Foundation of Jiangsu Province under Grant “Research and Realization on ASLR in operating systems”.

References


