Detecting Phantom Nodes in Wireless Sensor Networks ⋆

Joengmin Hwang, Tian He and Yongdae Kim

Department of Computer Science and Engineering, University of Minnesota

Abstract

In an adversarial environment, various kinds of security attacks become possible if malicious nodes could claim fake locations that are different from where they are physically located. In this paper, we propose a secure localization mechanism that detects the existence of these nodes, termed as phantom nodes, without relying on any trusted entities, an approach significantly different from the existing ones. The proposed mechanism enjoys a set of nice features. First, it does not have any central point of attack. All nodes play the role of verifier, by generating *local map*, i.e. a view constructed based on ranging information from its neighbors. Second, this distributed and localized construction results in quite strong results: even when the number of phantom nodes is greater than that of honest nodes, we can filter out most phantom nodes. Our analytical results as well as simulations under realistic noisy settings demonstrate our scheme is effective in the presence of a large number of phantom nodes.

Key words: Sensor networks, localization, secure localization, location verification, speculative algorithm, decentralized algorithm

⋆ The abbreviated version of this paper is presented in (16). This paper provides a set of extensions and improvements. We added the state of the art about the related works, and describe the limitation of conventional approach in detail. Importantly, a new atomic commit protocol is proposed to deal with collusion attack. We also reveal the detrimental effect of the collinear pivots. The analysis and extended evaluation is added.

Email address: jhwang,tianhe,kyd@cs.umn.edu (Joengmin Hwang, Tian He and Yongdae Kim).

1 Introduction

With thousands of tiny devices, Wireless Sensor Networks (WSNs) can support ubiquitous surveillance with a very low profile, and they can be quickly deployed without infrastructure. These features make them attractive for a wide variety of applications such as environmental and habitat monitoring (36), surveillance and tracking for military (13), emergency response and structural monitoring (40). Networked sensors can monitor the behavior of animals in wildlife. It may be used to detect smoke in forest to indicate a fire. The passage of soldiers or tanks of enemy can be monitored. In such mission, the essential query accompanying the detection of events is, where are the detected animals, where is the fire, where is the enemy? To answer this question, a sensor node needs to know its location in deployment area. When we are interested in only a certain area, we can query only nodes in a certain geographic area. In addition, knowledge of node location can be used to support various location aware application. For example, it can be used for location-based routing (19; 2) or geographic hash table (30). The wakeup schedule of sensor nodes can be more efficiently coordinated based on location of sensor nodes while providing enough sensing coverage (1; 9). The location of sensor node can be used to track the movement or behavior of targets (35; 28; 11). For example, in hospital we can track the movement and interaction of patients, doctors, nurses.

These applications run correctly when the localization error is limited to a certain range (12). However, if malicious nodes (attackers) distort the coordinate system severely by claiming fake locations, the performance of these applications could degrade significantly. To address these issues, various methods (8; 17; 18; 20; 22; 31; 37; 38) are proposed. They provide a set of nice mechanisms to detect and filter out compromised nodes and anchors. Most approaches depend on a few trusted entities (nodes or anchors), requiring at least the majority of these entities are not compromised. We argue that since the number of trusted entities in these approaches is relatively small, it would be relatively easy to break. Naturally, we raise the following question: *Is it possible to design a pure decentralized secure localization scheme that can detect phantom nodes, without requiring any trust entities?* The objective and intellectual contribution of this work lie in our answer to this challenging question.

1.1 Protecting Faked Ranging and Location

To obtain correct location information in the presence of attackers, several approaches have been proposed. These approaches share some common features:

adversaries try to fake location information and honest nodes (called verifiers) try to verify if each piece of location information is correct. Depending on which information the approach verifies and who plays the role of the verifier, there can be three different approaches.

In the first category, few trusted, but centralized nodes (called verifiers) validate the position or ranging claims of individual nodes (18; 31; 37). (We call this Centralized Phantom node Detection, *CPD* in short.) In *CPD*, a set of verifiers are effective to detect Sybil and Wormhole attacks independent from the number of attackers. However, it is not clear how to protect such verifiers from adversaries in WSNs. We argue that the *CPD* approach is relatively easy to break, because the number of verifiers is normally much less than that of regular nodes, and they could be traced and located more easily based on the traffic pattern, since they are involved with more communication than regular nodes.

In the second category, (set of) regular nodes verify the location information of anchors, who know their location. Through this verification, regular nodes can filter out compromised anchors or beacon information (possibly) generated by adversaries (20; 21; 22). (We call this Compromised Anchor Detection, *CAD* in short.) When a regular node receives a set of beacons from anchors in its range, it find the majority of consistent beacons. These approaches rely on the assumption that the majority of anchors or their beacons are not compromised. However, for a given area, if a majority of anchors are compromised *all* regular nodes in the area will be deceived. The number of anchors are relatively few in deployment area, and they are easy to be found and compromised by attacker because they periodically announce their locations.

A common weakness of the above two approaches comes from the centralization. A natural approach that overcomes this weakness is to remove the centralized nodes. In other words, it is desirable to design a localization mechanism where each node plays the role of both verifier and regular node, so that they can filter out phantom nodes. We call this new approach as Decentralized Phantom node Detection, *DPD* in short. *DPD* can inherently overcome some of the weaknesses of both *CPD* and *CAD*. Since *DPD* does not have trusted verifiers, attackers need to compromise much more nodes to make the attack successful. Furthermore, since there is no globally shared information, the damage, if any, is confined locally. We argue that a successful implementation of *DPD* would make localization much more robust against various attacks.

1.2 The Contributions of This Work

In this paper, we are targeting to the scenarios where attackers announce phantom nodes, who fake their locations, in proximity of legitimate nodes. To design a mechanism to support *DPD*, we focus on the development of the local map for individual nodes. A local map is a visual representation on the locations of neighbors of a node, which can be constructed correctly by verifying all location claims of its legitimate neighbors and filtering out phantom nodes generated by attacks.

Briefly, to find an actual local map without including phantom nodes, we project the neighboring nodes on a virtual plane and identify the inconsistency exhibited by the phantom nodes. Since there are no trusted entities, the process is *speculative* in nature. Interestingly, we demonstrate that this speculative process can filter the phantom node out with a very high probability when the process is repeated multiple times. In addition, since this novel speculative process mandates no agreements among neighboring nodes, it leads to two immediate benefits: First, the breach caused by the attacks is confined. A node's compromised decision does not propagate to affect other nodes' decisions. Second, much less information exchange is required, leading to less energy consumption, a nice property desired by WSNs. Beside these two benefits, our approach has the following major contributions:

- First, we propose an atomic commitment protocol to prevent phantom nodes generating consistent ranging claims.
- Second, our approach recovers a local map by projecting regular nodes at their locations and detect/filter phantom nodes. It requires no trusted beacons or anchors. We demonstrate that we can successfully filter out phantom nodes even when *the number of phantom nodes is much larger than the honest nodes*.
- Our approach works well under noisy distance measurements.

The remainder of the paper is organized as follows: Section 2 introduces the assumptions and notations. Section 3 provides an overview and the details of our approach are introduced in Section 4. The effectiveness of our approach on existing attacks is shown in Section 5. We present the experimental results in Section 6, and present related works in Section 7. Section 8 concludes the paper.

2 Preliminaries

In this section, we present our assumption used in the rest of paper. We assume all legitimate communication channels are established bidirectionally: if Node i can hear Node j, then Node j can hear Node i. In wireless networks, asymmetric links (41) are common due to the hardware calibration, the anisotropic property of the antenna and propagation media. However, bidirectional links can be easily established through a two-way handshaking. To make localized filtering effective, we also assume a reasonable network density (e.g., > 10 nodes per radio range). In the paper, for the sake of clarity, we describe the protocol in a two-dimensional plane. However, our approach can be applied to higher-dimensional spaces as well.

We assume range-based localization. A ranging is to measure physical properties such as distance or angle. In our work, we assume that the ranging is provided by each node based on the Time Difference of Arrival (TDoA) between RF signal and ultrasound signal (However, other different signals, or other ranging technique can be similarly applied). We briefly introduce the conventional localization based on TDoA which will be modified in our proposed approach. In the conventional TDoA, Node i generates RF signal and ultrasound signal simultaneously. Then, its neighbor j receives RF signal and ultrasound signal after certain propagation times, where the arrival of ultrasound signal is later than RF signal due to its longer propagation speed. As the propagation delay of RF signal is negligible, the time difference between the arrivals of RF signal and ultrasound signal can be used to estimate the distance between the two nodes. The measured distance $\hat{d}_{ij}$ is the time difference of arrivals of RF signal and ultrasound signal multiplied by the ultrasound propagation speed.

After ranging measurement, the positions of nodes are estimated by minimizing the differences between the measured distances and estimated distances. This process can be formally described as follows: Let $\tilde{d}_{ij}$ be the estimated distance between Nodes i and j. The sum of differences between the measured distances and the estimated Euclidean distances is:

$$\sum_{i=1}^{N} |\hat{d}_{ij} - \tilde{d}_{ij}|^2 \tag{1}$$

The position of Node j's location is estimated by taking Minimum Mean Square Estimate (MMSE) of the equation, following the method in (32). In graph theory, this is known as the graph realization problem, finding Euclidean positions for the vertices of a graph. According to (33), this is strongly NP-hard for the two-dimensional case or higher. Knowing the length of each graph edge does not guarantee a unique realization, because deformations can exist in the graph structure that preserve edge lengths but change vertex positions.

Rigidity theory distinguishes between non-rigid and rigid graphs. Non-rigid graphs can be continuously deformed to produce an indefinite number of different realizations, while rigid graphs cannot.

Our proposed scheme is suitable to both localization with and without anchors. Without anchors, the global location can be still obtained by methods in (23; 10). In localization with anchors, we can obtain the global location directly from anchors while detecting the compromise of anchor as described in Section 5. In this case, the strength of security increases proportional to the number of neighbors including anchors and regular nodes.

2.1 Notation

The following notations are used throughout the paper.

- v: a node which verifies the locations of its neighbors
- $Nbr(v)$: the node set consisting of v's neighbors and v
- p_k: the location of node k on virtually computed local plane
- N: the number of neighboring nodes
- M: the number of inter-node distance measurements
- d_{ij}: the **physical** distance between nodes i and j
- $\hat{d}_{ij}$: the **measured** distance to node j by i
- $\tilde{d}_{ij}$: the **computed** distance between nodes i and j
- D: a set of distance measurements, i.e., $\{\hat{d}_{ij} \mid i, j \in A\}$

3 Overview

Here, we provide an overview on our two-phase approach to detect the phantom nodes. First, in Section 3.1, we explain how to prevent the phantom nodes generating consistent ranging (distance) claims [1] to multiple honest nodes. Second, if the phantom nodes exhibit a set of inconsistent ranging claims, we propose a speculative method to detect them in Section 3.2.

[1] Here, a set of ranging claims is said to be consistent, when such claims can project a node into a physical location in the 2-D or 3-D space (in case of 3-D localization) and the distances between this physical location and other nodes' locations match the claims.

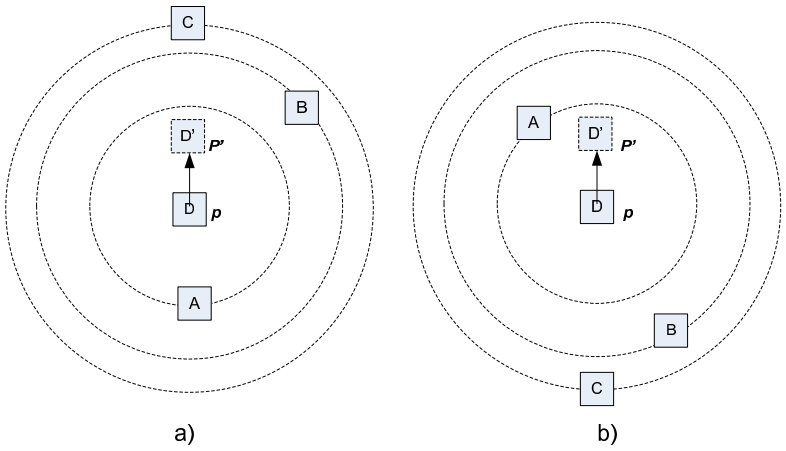

Fig. 1. The difficulty in generating consistent range claims

3.1 Prevent Faked Consistent Ranging Claims

Our basic design only allows a node to claim about its *distances* to other neighboring nodes, not its own *location*. Therefore, to disrupt the operation of location-dependent applications, (a set of) malicious nodes, whose goal is to create a phantom node, must fake a set of distances to all of its neighboring nodes. If the locations of neighboring nodes are known *a priori*, a set of fake, albeit consistent, ranging distances can be easily created by calculating the distances from a fake location to each of its neighbors' location. Therefore, it is important for our design to *hide* the location information during the ranging process. Without the location information of the neighboring nodes, it is hard for an attacker to generate a set of consistent ranging values (distances), and hence to fake itself into a different physical location. For example, as shown in Figure 1a, suppose an attacker D at the location p obtains three ranging distances in the 2-D space from three honest nodes A and B and C, it can only conclude that A, B and C are located at the edges of three concentric circles centered at p. To claim a different physical location p' within the 2-D space, the attacker D needs to fake three different ranging distances that are consistent. Without knowing the precise locations of the neighbors, such consistency is difficult to achieve. As shown in Figure 1a, to move from the position p to p', the attacker D needs to claim two shorter ranging distances to Nodes B and C, but a longer ranging distance to Node A; However in case of Figure 1b, the attacker D needs to claim the opposite. Since the locations of A, B and C are *unknown* to the attacker, it cannot decide which claim to make. We note that a sensor network normally has a high node density ($\gg 10$), which makes a consistent ranging claim practically impossible without the neighbors' location information.

We note that the aforementioned basic design works correctly if the attackers do not collude with each other. In the case of collusion, it is possible that in 2-D

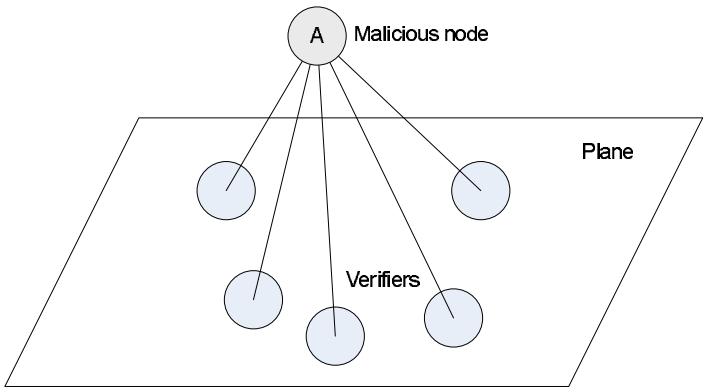

Fig. 2. A fake location with inconsistent ranging distances

case, multiple attackers collaboratively identify the locations of honest nodes. Specifically, if three attackers can obtain the distances between themselves and a honest nodes, the location of this honest node can be calculated using the trilateration technique (23). Once the locations of honest nodes are obtained, the attackers can generate consistent, albeit faked, range claims. The key idea to deal with colluded attack is to reveal all ranging values *atomically* by using an atomic commit protocol, which we describe in more detail in Section 4.1

In summary, to prevent phantom nodes generating a set of fake, albeit consistent, ranging claims, the design guideline is *hiding the location and ranging distance information before revealing them atomically*. Once the consistent ranging claims are prevented, we can identify the phantom nodes by detecting the inconsistent ranging claims, which is addressed in the rest of the paper.

3.2 Detect Faked Inconsistent Ranging Claims

The goal of attackers is to convince honest nodes the existence of several non-existing nodes or the fake locations of the attackers. To achieve this, attackers generate fake ranging information between the phantom nodes and other honest nodes. Figure 2 illustrates this idea. An attacker claims that it has a set of distances to all neighbors in order to fake its position. However, the design guideline aforementioned ensures that such a claim is *inconsistent* with high probability. In other words, there is no such location in the plane that satisfies these distance constraints simultaneously. This can be easily explained in visualization method using classical multidimensional scaling (34). In this method, a set distance measurements between any two neighbors are taken for principal coordinate analysis. From the analysis it creates a configuration of points. Ideally, those points are in two or three dimensions, and Euclidean distances between them reproduce the original distance measurements. However, if a part of the distance measurements are compromised, the points are reproduced in higher dimensional space.

With trusted verifiers equipped with ultrasound devices, the distance bounding method (31), can easily reject the distance claims that are shorter than the real distance. However, these trusted verifiers could be an attractive target of attackers. In the second part of the paper, we propose a speculative method to detect such inconsistency in phantom nodes in sensor networks. More specifically, our objectives are (1) to detect phantom nodes that have inconsistent ranging claims, (2) to identify the real location of legitimate nodes, (3) potentially to find the positions of attackers that are generating phantom nodes.

4 The Detailed Approach

In this section, we focus on identifying the phantom nodes that generate inconsistent ranging from/to the set of honest nodes. For simplicity, we describe a two-dimensional localization. However, our algorithm extends straightforwardly to three dimensions.

Formally, we stated the problem as follows:

Definition: A set of nodes is *consistent*, if and only if the nodes can be projected on a single Euclidean plane (in 3-D case, Euclidean space), keeping the measured distances among themselves.

Problem: Given a node set $Nbr(v)$ that consists of a node v and its neighbors, and a distance set D that consists of the measured distance, denoted by $\{\hat{d}_{ij} | \hat{d}_{ij} = \hat{d}_{ji}, i, j \in Nbr(v), i \neq j\}$, find the largest consistent subset of $Nbr(v)$.

We divide the algorithm into two main phases: distance measurement and phantom node filtering. In the first phase, each node measures the distances securely to its neighbors using an atomic commit protocol. In the second phase, each node projects its neighboring nodes to a virtual local plane to determine the largest consistent subset of nodes. After two phases are complete, each node establishes a local view without phantom nodes. Such a local view is useful in many services such as location-based routing and sensing coverage. Alternatively, any local coordinate system can be reconciled into a unique global coordinate system. The algorithm is easily distributed because a node only uses distance measurements to immediate neighbors and between neighbors. Furthermore, if one node in the network moves, only the $O(1)$ clusters containing that node must update their position information. The following two sections describe the phases of the approach in more detail.

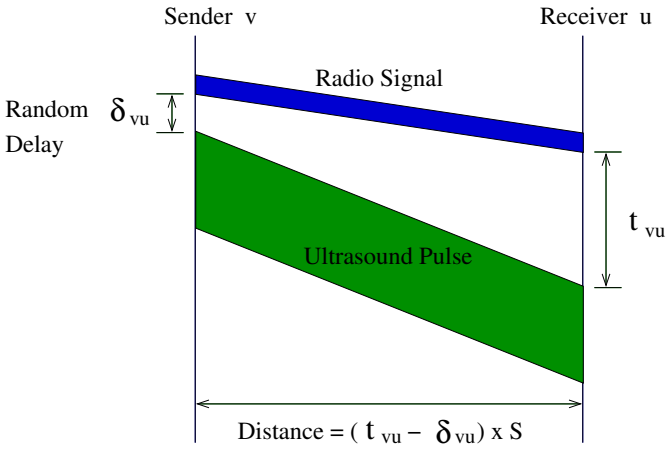

Fig. 3. Ranging Process in Phase 1

4.1 Phase I: The Atomic Commit Protocol for Distance Measurement

We note that it is possible that an attacker could silently collects the distance announcement from honest nodes, and then calculates the relative locations of the honest nodes. As a result, this attacker could fake a set of consistent range claims later on. To prevent such type of attack, we require that distance measurements are announced in an atomic manner. This is achieved by our atomic commit protocol. This protocol is generic enough to apply to diversified ranging technique. However for the sake of clarity, here we describe an instance that is based on the Time Difference of Arrival (TDoA) technique (32), which estimates the distance using the difference of propagation speeds between radio and ultrasound signals

Specifically, our atomic commit protocol consists of two steps:

Step 1: Disseminate Encrypted Measurements

(1) Each node v announces that it will participate in the secure localization. Let $Nbr(v)$ be the set of all neighbors of v announcing participation.
(2) Node v first sends an RF signal to a node $u \in Nbr(v)$. After a random delay δ_{vu} v also sends ultrasound signal to the node u, while keeping the delay δ_{vu} secret. Receiver node u records the time t_{vu} until ultrasound signal arrives after it receives RF signal.
(3) After collecting ultrasound signal and RF signal from every node in $Nbr(v)$ (or after pre-determined time interval), v encrypts δ_{vu}, t_{uv} for every $u \in Nbr(v)$ using a secure symmetric key encryption algorithm with a fresh random key k. Node v, then, broadcasts this information to its neighbors. Any node failing to report in this step will be dropped from $Nbr(v)$.

Step 2: Reveal Measurements Atomically

(1) After collecting encrypted messages from every node in $Nbr(v)$ (or after pre-determined time interval), v discloses the encryption key k. Any node that fails to reveal its encryption key in this step will be dropped from $Nbr(v)$.
(2) After received encryption keys from every node in $Nbr(v)$, v can compute the distance $\hat{d}_{ij}$ between any node i and j in $Nbr(v)$. As propagation delay of RF signal is negligible, the distance can be estimated as $(t_{ij} - \delta_{ij}) \times S$ as shown in Figure 3, where S is propagation speed of ultrasound. Note that 1) t_{ij} are included in the message from j and δ_{ij} is included in the message from i.
(3) After collection of neighbors' announcements, the node v compares the data collected. For each collected distance, if $\hat{d}_{ij} = \hat{d}_{ji}$, it is included in the filtering phase which is described in Section 4.2.

Note that this algorithm prevents a node u from computing the location of v, since, even after receiving the ultrasound and RF signal from v, u cannot correctly compute the distance to v due to the random delay. Furthermore, v's information is disclosed only after every node in $Nbr(v)$ announces its ranging information in step 1. After disclosing the symmetric key in step 2 atomically, nodes in $Nbr(v)$ are able to compute the distance to v. At this point of time, malicious node can no longer fake new distance measurement.

We employ a symmetric key to make the encryption and decryption cheap. The ranging information needs to be encrypted to make the information *secret* until disclose step. The ranging information also needs to be *verified* that it is generated by the node that really generated it. The symmetric key can be used to satisfy both purposes. The key does not need to be pre-distributed before deployment. This is because although an attacker generates its key on-line randomly, it cannot generate consistent ranging unless it knows other honest nodes' location. It only increases the number of phantom nodes participating.

Our approach can be used based on other localization technique other than TDoA. For instance, if the distance is measured by Received Signal Strength Indication (3; 4) instead of time difference of arrivals between ultrasound signal and RF signal, we only need to change following part in the protocol described above. In Step 1, node v sends an RF signal to a node $u \in Nbr(v)$ with random signal strength setting s_{vu} while keeping the setting secret, instead of using random delay δ_{vu} in the above. Receiver node u records received signal strength s'_{vu} instead of recording the TDoA between ultrasound signal and RF signal, t_{vu} in the above. In Step 2, it computes the distance $\hat{d}_{ij}$ between any node i and j in $Nbr(v)$ using radio propagation model, signal strength at sender, and the received signal strength.

Algorithm 1. Speculative filtering

for $i = 1$ to *iter* **do**
  each node v picks up two neighbors i and j randomly
  create local coordinate system L using $v, i, j, \hat{d}_{vi}, \hat{d}_{vj}, \hat{d}_{ij}$
  initialize undirected graph $G(V, E)$
  for each neighbor $k \in Nbr(v)$ **do**
    calculate the location of k, p_k, on L by multilateration of $\hat{d}_{kv}$, $\hat{d}_{ki}$ and $\hat{d}_{kj}$ from v, i, j
  end for
  create node v in V with location p_v
  for each neighbor $k \in Nbr(v)$ **do**
    create node k in V with location p_k
  end for
  for each pair of nodes $i, j \in V$ and their ranging $\hat{d}_{ij}$ **do**
    $\tilde{d}_{ij} = |p_i - p_j|$
    if $|\hat{d}_{ij} - \tilde{d}_{ij}| < \epsilon$ **then**
      create edge $e(i, j)$ in E
    end if
  end for
  find the largest connected cluster C and save it
end for
Among all saved C, choose the one with the largest size

In this section, we propose a novel speculative procedure, which can effectively and efficiently filters out phantom nodes. The filtering procedure is described in Algorithm 1. Initially, the node v picks up two neighbors i and j randomly as pivots. (Note that node i and j could be phantom nodes themselves). Using the node v as the origin, the neighbors i and j and three distance measurements among v, i and j, the local coordinate system L is constructed. In the node v's coordinate system, we use a graph $G(V, E)$ to construct a consistent subset. The set V is used to contain the node v and its neighbors, and the set E is used to keep the edges between two nodes when the distance information between them maintains consistency. Initially the graph G is empty. The update process of the graph G is as follows: The location of the neighbor k is determined on the local coordinate system L by trilateration (32) from three nodes v, i, j with measured distances $\hat{d}_{kv}$, $\hat{d}_{ki}$ and $\hat{d}_{kj}$. The relative locations of the four nodes v, i, j, k are unique to a global rotation, translation, and reflection. In graph theory, the quadrilateral is globally rigid.

After projecting all the neighbors on L, the distance between the projected neighbors is compared with the measured distance. For any two nodes i and j the distance $\tilde{d}_{ij} = |p_i - p_j|$ is calculated from the projected location on L. If $|\hat{d}_{ij} - \tilde{d}_{ij}| \geq \epsilon$, the edge between i and j is not included in E. (the threshold

value of ϵ depends on the noise of the ranging results. More information can be found at Section 6.) The largest connected set V that contains node v is regarded as the largest consistent subset in the speculative plane L. This filtering procedure is done *iter* times (*iter* is a key parameter discussed later in Section 4.5), and the cluster with the largest size is chosen as a final result.

4.3 Identifying Consistent Subset

Algorithm 1 obtains a connected cluster in each iteration. In this section, we show that (i) the one with the largest size must consist of only legitimate nodes and (ii) we can determine the case where a chosen pivot is, unfortunately, a phantom node.

Theorem 1: *When all the pivots chosen are honest nodes, the consistent cluster computed by the proposed solution in Algorithm 1 does not contain phantom nodes.*

Proof Sketch: Let Node v at location p_1 creates a local map. It selects two other pivots at locations p_2, p_3. We denote the plane created by honest pivots at locations p_1, p_2, p_3 by P and the largest cluster by C. Obviously, every honest node will be projected to real plane P, keeping the consistent relative positions. As a result, Algorithm 1 creates the edges between honest nodes. On the other hand, in our attack model, on behalf of a phantom node, the attacker generates a set of distances. According to Section 3.1, these distances cannot be projected consistently on the real plane P. Instead such a phantom node is projected to a point on the phantom plane P' as shown in the Figure 4. Therefore, the consistent edges between phantom node and honest node can not be created, if without coincidence. Therefore, the consistent cluster with honest pivots at locations p_1, p_2, p_3 is composed of honest nodes.

Theorem 2: *If at least one of pivots is a phantom node, the size of largest cluster is small compared to the one when none of pivots is a phantom node.*

Proof Sketch: Let Node v at location p_1 creates a local map. It selects two other pivots at locations p_2, p_3, at least one of which is a phantom node. Say pivot at location p_2 is a phantom node. We denote the plane created by pivots at locations p_1, p_2, p_3 by P' and the largest cluster by C'. From ***Theorem 1***, we know the phantom pivot at location p_2 is not on plane P . Thus, the plane P' containing p_2 is different from P. Since P' is different from P, any node on P cannot be on P' unless it is located on the line, the intersection of P and P' as shown in Figure 4. As a result, most of legitimate nodes are excluded from C'. Since the filtering is local process of legitimate node v, and pivoting is speculatively determined by v, *the inconsistent distance measurements project the non-collusive phantom nodes on different phantom planes.* Thus, with a

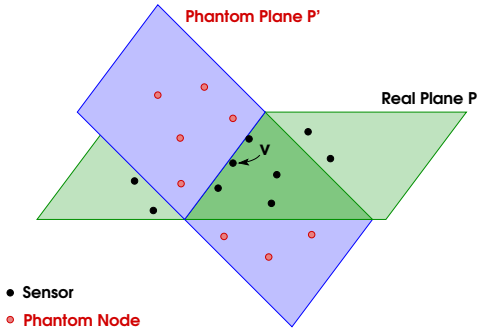

Fig. 4. Real plane vs. phantom plane

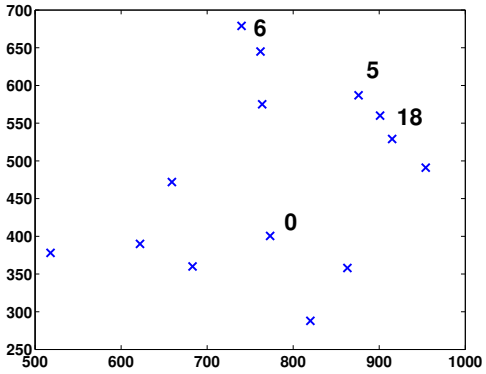

Fig. 5. Real plane

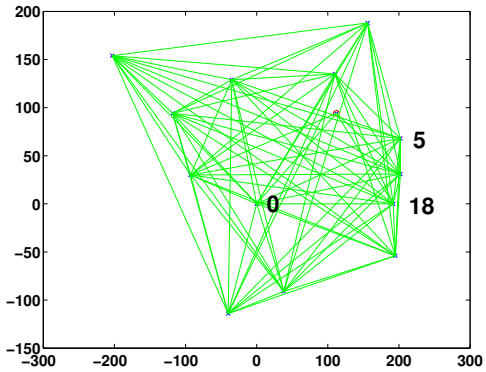

Fig. 6. Computed plane from pivot 0,5,18

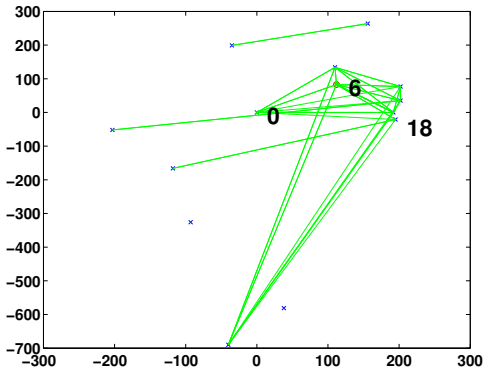

Fig. 7. Computed plane form pivot 0,6,18

high probability, the consistent edge between phantom nodes and chosen pivots can not be created. This leads any cluster C' made by the phantom nodes on a phantom plane P' is much smaller than C on P.

Case Study: As an example, Figures 5, 6 and 7 reflect the properties of ***Theorem 1 and 2***. Figure 5 plots the real locations of the nodes, among which node 0 is a verifying node, node 6 is a phantom node, node 5 and 18 are not compromised, Figure 6 shows the cluster created when the pivot (node 5 and 18) is not compromised (Theorem 1), Figure 7 is the cluster when the phantom pivot (node 6) is used, which size is much smaller than the size of cluster shown in Figure 6 (Theorem 2).

4.4 Localized Adversarial Effect of a Phantom Plane

In Section 4.3, due to the speculative nature of our approach, we assume that with a high probability, a large phantom plane can not be created consistently among non-collusive phantom nodes. In this section, we study the impact if this assumption doesn't hold, i.e., phantom nodes are able to launch collusion attack. As shown in the Figure 4, node v is located at the intersection of

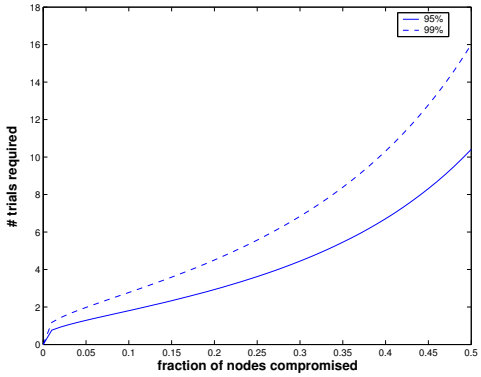

Fig. 8. The number of trials required to ensure the correct pivoting

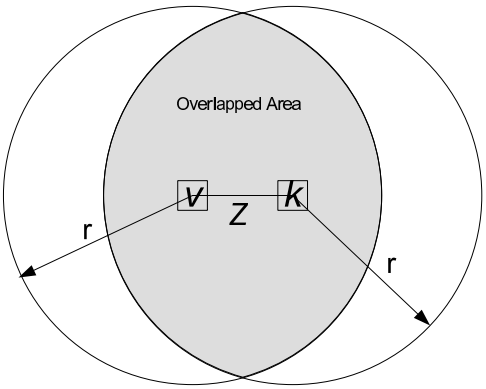

Fig. 9. The overlapped area

a phantom plane and a real plane. If the number of phantom nodes on the phantom plane is larger than the number of node on the real plane, node v will be deceived. It is also true that any other nodes located in the intersection line can be deceived as well. However, we note that any honest node that is not located at the intersection, can not *perceive* the existence of the phantom plane P' from *its own local view*. It can only *perceive* the large real plane P. *This observation indicates that even if a large phantom plan is created through collusion attack, it can only compromise the views of a limited number of nodes located at the intersection line.* Suppose there are N honest nodes in the real plane P and none of three nodes are collinear, to compromise the views of all these nodes, more than $\frac{N^2}{2}$ phantom nodes are needed, which is much larger than the number of honest nodes N. This gives us the insight why our scheme works well in the presence of a large number of phantom nodes.

4.5 Cost Analysis

The cost for our protocol consists of (i) the communication cost during exchange of distance information in Phase I, (ii) computation cost for encryption and decryption of symmetric key, and (iii) the computation cost in filtering phase.

In our proposed solution, suppose a node has $N-1$ neighbors. It generates ranging information with $N-1$ neighbors, thus communication cost is $O(N)$.

Each node encrypts its own distance measurement message, and decrypts neighbors distance measurement message. Thus, the the computation cost for encryption and decryption of symmetric key is $O(N)$.

The computation cost in filtering phase for each node is estimated as follows: The projection cost by trilateration is proportional to the number of neighbors $N-1$. The projection is done for several trials. From Algorithm 1, we know

that *iter* controls the number of trials. Thus, the computation cost for each node is $iter \times (N-1)$. The value *iter* is determined differently depending on the ratio of honest nodes and phantom nodes. This is because our algorithm is speculative in nature. Obviously, it is unacceptable if such speculation takes a large number of trials.

We'll show the expected number of trials *iter* in Algorithm 1 is small even with a large percentage of phantom nodes. To identify the largest consistent subset, our speculative algorithm can not stop before the node v successfully selects two honest pivots. Suppose q is the probability that a random neighbor is a phantom node, the probability at least one of pivot is a phantom pivot is $1-(1-q)^2$. During *iter* trials, the probability that at least one of trial succeeds in selecting two honest pivots is:

$$P[X \geq 1] = 1 - (1 - (1-q)^2)^{iter}$$

The number of trials required to ensure the successful filtering with probability 95% and 99% is shown in Figure 8. As shown in Figure 8, even 50% nodes are phantom nodes, the number of trials needed is only 16 to achieve 99% success ratio. We also note that the number of trials only affects the computation overhead. No extra communication is needed when the number of trials increases.

With the node density of 10, to achieve 99% success ratio, we need 16 iterations with 10 trilateration per iteration, a computation that finishes within several milliseconds in Mica motes.

4.6 Number of Trials Available

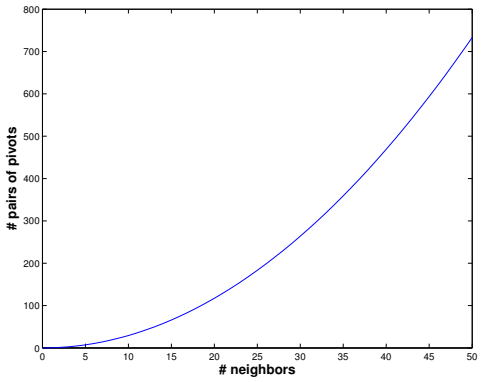

Fig. 10. Average number of possible pairs of pivots for the given number of neighbors

If the computation overhead can be ignored, ideally we want to perform as many trials as possible. This section analyzes the expected number of trials

available for computation. Suppose the node v can obtain distance measurements from all neighboring nodes that are less than r meters away, as shown in Figure 9, to verify a certain node k we need at least two honest nodes exist within common range r from v and k so that an unique plane is determined. And, as long as there exist at least two honest nodes within the overlapped area shown in Figure 9, each local unique plane can be stitched together, resulting in one unique plane with v at the origin.

As described in Algorithm 1, a node v uses itself as the first pivot, and two randomly selected neighbors are used as second and third pivot. For one test node k, if z is the distance between node v and k, the size of overlap region shown in Figure 9 is:

$$A_{overlap}(z) = 2r^2 cos^{-1}(\frac{z}{2r}) - z\sqrt{r^2 - (\frac{z}{2})^2}$$

Two selected pivots must be the common neighbor of node v and the the tested node k. The average number of common neighbors throughout all the possible values of z is:

$$N_{avg} = \frac{n}{\pi r^2}\int_0^r f(z) \times A_{overlap}(z)dz = 0.586503 \times n$$

where $f(z)$ is probability density function of z.

$$f(z) = \frac{\partial[Pr(Z \leq z)]}{\partial z} = \frac{\partial}{\partial z}[\frac{\pi z^2}{\pi r^2}] = \frac{2z}{r^2}.$$

Since the average total number of common neighbors is $0.586503 \times n$, the number of available pairs is approximately $((0.586503 \times n) - 1) \times (0.586503 \times n)/2$. As we know from Section 4.5, 16 trials can render a 99% success ratio even with 50% phantom nodes. Accordingly, a node density of 10 neighbors can sufficiently satisfy the requirement. To illustrate the analytical results numerically, Figure 10 shows the relation between the number of neighbors and the number of pairs available for speculative filtering.

4.7 Avoid Collinear Pivots

If two pivots chosen are collinear with node v, i.e., they form a straight line, the chance that a node's location flips from one side of this line to the other side is high. This type of localization error could lead to a high rate of false positives in which an honest node are filtered out as phantom nodes. In this section, we investigate the relation between ranging measurement error and localization error in cases of collinear and non-collinear pivots. Our conclusion

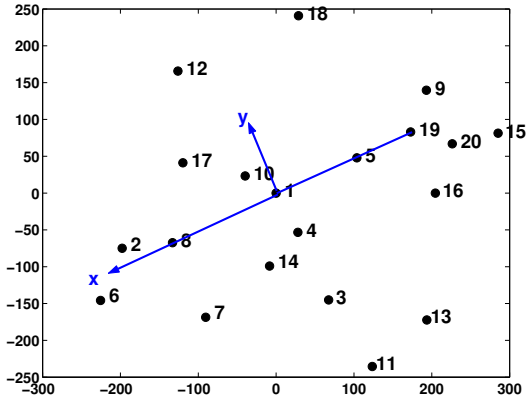

Fig. 11. Collinear pivots case

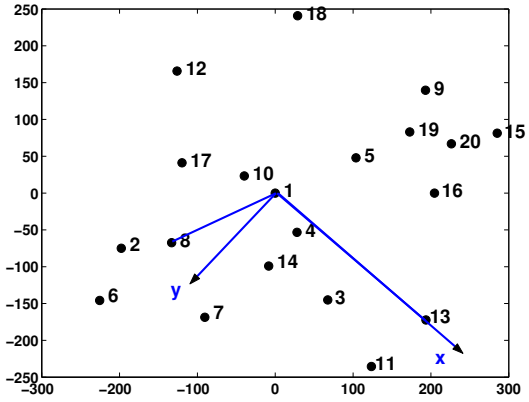

Fig. 12. None collinear pivots case

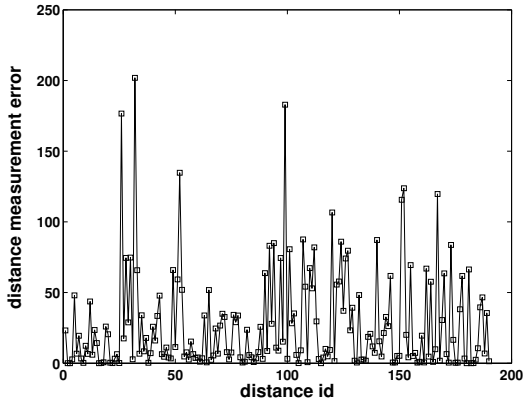

Fig. 13. Distance measurement error

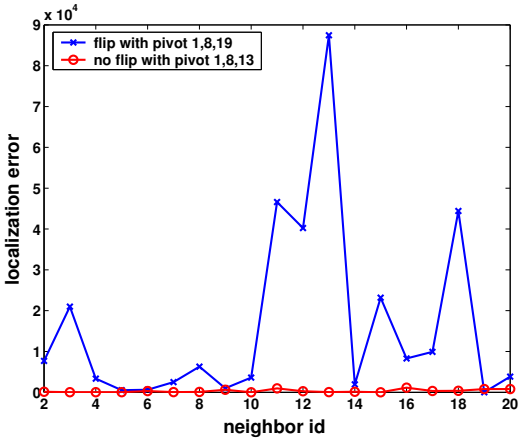

Fig. 14. Localization error

from this study indicates that it is critical for us to prevent using collinear pivots to establish the local coordinate system. Following are details of our analysis.

One criterion by which we evaluate the performance of the algorithm is how the computed localization differs from the known ground truth. This error is expressed as:

$$\sigma_p^2 = \sum_{i=1}^{N} \frac{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}{N}$$

where N is the number of nodes, $\hat{x}_i$ and $\hat{y}_i$ compose the localized location of node i, and x_i and y_i composed the true location of node i. This metric is simply the mean-square error in Euclidean 2D space.

It is useful to compare σ_p^2 to σ_d^2, the mean-square error in the raw distance measurements, since the error model of the measurements determines the minimum achievable σ_p of an ideal localization algorithm. The mean-square error of the distance measurements is:

$$\sigma_d^2 = \sum_{i,j} \frac{(\hat{d}_{ij} - d_{ij})^2}{M}$$

where M is the number of inter-node distances, $\hat{d}_{ij}$ is the measured value of distance i, and d_{ij} is the true value of distance between node i and j.

Figures 11 and 12 show an example of neighbor distribution. Node v has node $id = 1$ and neighbor $2, \ldots, 20$ has real relative location shown in the figure. The distance measurement error for each pair of nodes is calculated as $(\hat{d}_{ij} - d_{ij})^2$ and shown in Figure 13.

In the first trial, pivots $1, 8, 19$ are selected and in the second trial, pivots $1, 8, 13$ are selected. In Figure 11, the pivots $1, 8, 19$ are collinear, which is prone to flipping. Figure 14 shows the comparison of the localization errors, $(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2$, for each node between first and second trail. Obviously the collinear pivots (Figure 11) cause significantly higher localization error than non-collinear pivots (Figure 12). In the collinear case, the large localization error is mostly from the vertex flipping rather than the ranging measurement error. To reduce such the error due to vertex flipping, we adopt the solution proposed in (23). According to (23), their algorithm regards only those triangles with a sufficiently large minimum angle as robust. To select the robust triangles, (23) chooses a threshold d_{min} based on the measurement noise and identify those triangles that satisfy: $b \sin^2 \theta > d_{min}$ where b is the shortest side and θ is the smallest angle. The equation bounds the worst-case probability of a flip error for each triangle. When d_{min} was chosen to be 3σ, for Gaussian noise, this bounds the probability for flipping to be less than 1%. In our solution, for a triangle made by three pivots and the shortest side b and smallest angle θ of the triangle the threshold is tested. The pivots that do not satisfy the condition will be excluded.

5 Robustness against Existing Attacks

In this section, we applies our proposed approach to existing attacks, i.e., wormhole attack, node replication, sybil attack, etc. We first create virtual planes following the approach described in the previous sections. Then, two virtual planes are stitched together using Algorithm 2. During this process, wormhole attack, node replication, sybil attack are detected.

The stitching process of two planes is described in Algorithm 2. It tests if a set of nodes C_1 on plane P_1 and another set of nodes C_2 on plane P_2 can be combined consistently. If there are more than three common nodes in C_1 and C_2, we fix those common nodes. If the relative topology of those nodes on P_1 are

Algorithm 2. Combine(P_1,P_2)

plane P_1 contains a set of nodes C_1 and plane P_2 contains a set of nodes C_2
if $|C_1 \cap C_2| < 3$ **then**
  return false
end if
for nodes $i, j \in C_1 \cap C_2,\ i \neq j$ **do**
  if P_1 and P_2 have different values for $\tilde{d}_{ij}$ **then**
    return false
  end if
end for
fix $C_1 \cap C_2$, and stitch two planes P_1 and P_2.
if There exists $\hat{d}_{ij}$, $i \in P_1, j \in P_2$, and $|\hat{d}_{ij} - \tilde{d}_{ij}| \geq \epsilon$ **then**
  return false
end if
return true

different from the topology on P_2, two planes cannot be combined consistently. When two planes are stitched together by fixing those common nodes, if there exists distance measurement $\hat{d}_{ij}$ between a node i in C_1 and another node j in C_2, and the difference between measured distance and computed distance on the stitched map is non-trivial, $|\hat{d}_{ij} - \tilde{d}_{ij}| \geq \epsilon$, they cannot be combined consistently. Otherwise, the stitched plane is regarded as a consistent map. In addition, if $C_1 \cap C_2 = C_1 = C_2$, then we conclude that C_1's map is the same as C_2's.

Figure 15 lists five types of phantom node attacks and our approach to detect the attacks. We describe the process in node 1's local view. The real plane in the first column in Figure 15 is the actual placement of each node. We mark the malicious nodes with circles. For each neighboring node or the nodes connected through wormhole, the distance measurements are collected as specified in the Section 4.1. From the collected distance measurements, node 1 creates its virtual plane as described in Algorithm 1. Then, we input the virtual plane to Algorithms 2.

(a) **No Attack**: In Figure 15a, the real plane consists of 9 honest nodes. Node 1 has distance measurements with nodes $2, \ldots, 9$. For any two planes P_1 and P_2 created by different sets of pivots, the same plane is created, and Algorithm 2 always returns true.

(b) **Compromised Anchor**: The anchor generates ranging information like other non-anchor nodes, and the compromised ranging information can be filtered with our proposed method. In addition, anchor generates beacons that contain anchor's location on global coordinate system. This information may not needed in fact because the global location can be always obtained by stitching local maps following methods in (23; 10). However, anchor nodes are provided, we can still detect the compromise of global location in beacon by comparing distance between global locations of two anchor nodes with the distance between two anchor nodes in the stitched local maps.

In our example in Figure 15b, the real plane consists of 10 regular nodes and two anchor nodes 11 and 4, where the anchor 4 is compromised. We assume the compromised anchor 4 generates consistent distance measurement, but modified anchor location in its beacon. The node 1 has distance measurements with $2, \ldots, 9$, and the node 10 has distance measurements with $2, 3, 10, 5, 6, 11, 8, 9, 12$. Two distinct virtual planes P_1 and P_2 are constructed by node 1 and node 10, and inputed to Algorithm 2. It returns true because the compromised anchor does not generate inconsistent distance measurement. However, when the distance between anchors on the stitched map, $\tilde{d}_{4,11}$, is compared with the difference of global locations, $|(x_4, y_4) - (x_{11}, y_{11})|$, we detect the inconsistency.

Real Plane	Distance Measurement	Virtual Plane after Filtering	Attack Detection
(a) No Attack 1 2 3 4 5 6 7 8 9	Node 1 collects: $\hat{d}_{1,2}, \ldots, \hat{d}_{1,9}$ $\hat{d}_{2,1}, \ldots, \hat{d}_{2,9}, \ldots$	Plane P_1 by node 1: 1 2 3 4 5 6 7 8 9	$C_1 \leftarrow \{1, \ldots, 9\}$ $C_2 \leftarrow \{1, \ldots, 9\}$ Combine$(P_1, P_1) \rightarrow$ true
(b) Compromised Anchor 1 2 3 10 4 5 6 11 7 8 9 12	Node 1 collects: $\hat{d}_{1,2} \ldots, \hat{d}_{1,9}, \ldots$ *beacon* : (x_4, y_4) Node 10 collects: $\hat{d}_{10,2}, \hat{d}_{10,3}, \hat{d}_{10,5}, \ldots$ *beacon* : (x_{11}, y_{11})	Plane P_1 by node 1: 1 2 3 4 5 6 7 8 9 Plane P_2 by node 10: 2 3 10 5 6 11 8 9 12	$C_1 \leftarrow \{1, \ldots, 9\}$ $C_2 \leftarrow \{2, 3, 5, 6, 8, 9, 10, 11, 12\}$ Combine$(P_1, P_2) \rightarrow$ true $\tilde{d}_{4,11} \neq \lvert(x_4, y_4) - (x_{11}, y_{11})\rvert$
(c) Wormhole Attack 1 2 3 11 12 13 4 5 6 14 15 16 7 8 9 17 18 19	Node 1 collects: $\hat{d}_{1,2}, \ldots, \hat{d}_{1,9}$ $\ldots, \hat{d}_{5,15}, \ldots$	Plane P_1 by node 1: 1 2 3 4 5 6 7 8 9 Plane P_2 by node 11: 11 12 13 14 15 16 17 18 19	$C_1 \leftarrow \{1, \ldots, 9\}$ $C_2 \leftarrow \{11, \ldots, 19\}$ Combine$(P_1, P_2) \rightarrow$ false
(d) Replication Attack 1 2 3 11 12 13 4 5 14 15 6 7 8 16 17 18	Node 1 collects: $\hat{d}_{1,2}, \ldots, \hat{d}_{1,8}$ $\hat{d}_{1,11}, \ldots, \hat{d}_{1,18}$ $\ldots$	Plane P_1 by node 1: 1 2 3 4 5 6 7 8 Plane P_2 by node 11: 11 12 13 14 15 16 17 18	$C_1 \leftarrow \{1, \ldots, 8\}$ $C_2 \leftarrow \{11, \ldots, 18\}$ Combine$(P_1, P_2) \rightarrow$ false
(e) Sybil Attack 1 2 3 4 5 6 7 8 9	Node 1 collects: $\hat{d}_{1,2}, \ldots, \hat{d}_{1,8}$ $\hat{d}_{1,11}, \ldots, \hat{d}_{1,18}$ $\ldots$	Two virtual planes P_1 and P_2 by node 1: 1 2 3 11 12 13 4 6 14 15 7 8 9 16 17 18 5 11 12 13 14 1 2 3 4 5 6 7 8 9 15 16 17 18	$C_1 \leftarrow \{1, \ldots, 9, 11, \ldots, 18\}$ $C_2 \leftarrow \{1, \ldots, 9, 11, \ldots, 18\}$ Combine$(P_1, P_2) \rightarrow$ false

Fig. 15. Attack detection

(c) **Wormhole Attack**: The wormhole attack is the attack where two nodes physically or logically located far away behave as if they have short path to deliver messages. Figure 15c shows wormhole attack. In the figure, 9 nodes are located far away from other 9 nodes. Thus, the node 1 has distance measurement only with $2, \ldots, 9$ because it is far away from other 9 nodes.

Two compromised nodes in the center create wormhole to pretend to be neighbor. For example, node 5 announces a fake distance measurement with node 15. Two virtual planes P_1 and P_2 created by node 1 and 11 are failed to be combined in Algorithm 2 because distance measurement between node

5 and 15, $\hat{d}_{5,15}$, is not matched to the distance $\tilde{d}_{5,15}$ on the stitched map.

(d) **Replication Attack**: Figure 15d shows the replication attack through a wormhole, in which 8 nodes are located far away from other 8 nodes. The attacker does not involve in any direct compromise of nodes, but two replicators in the left and right side each, replicate the messages of honest nodes through the wormhole.

For example, a replicator in the left side replicates distance measurement messages of nodes $1, \ldots, 8$ in the left side to the other replicator in right side which announces the replicated messages to nodes $11, \ldots, 18$ in right side. The distance measurement messages of nodes $11, \ldots, 18$ are also replicated from right side of nodes to left side of nodes in the similar way. In result, nodes in left side and nodes in right side believe they are neighbors to each other. Node 1 creates a plane P_1 filtering inconsistent distance measurement between nodes in left side and in the right side. In result, the plane P_1 contains nodes $1, \ldots, 8$. Node 11 creates a plane P_2 filtering inconsistent distance measurement between nodes in left side and in the right side. In the result, the plane P_2 contains nodes $11, \ldots, 18$. Two planes P_1 and P_2 fails in Algorithm 2 because they do not have common nodes.

(e) **Sybil Attack**: The sybil attack is the attack where a compromised node takes role of several node entities to the honest node, thus consume the fixed resource more than it has to. Figure 15e is when the sybil nodes are created. The node in the center is compromised and it takes a role of non-existing node $11, \ldots, 18$. If it inserts false distance measurement messages between phantom nodes $11, \ldots, 18$, it may create phantom plane consisting of phantom nodes. However, the honest nodes on real plane P_1 will be separated from the phantom nodes on the phantom plane P_2. Two planes P_1 and P_2 fails in Algorithm 2.

6 Simulation Results

In this section, we provide the simulation results for our proposed scheme. We generate a set of sensor nodes on the random locations. The node and all of its neighbors participate in the phantom node detection in a decentralized manner. After each node updates its neighbor list, they exchange distance information and filter phantom nodes. We describe the performance metric and simulation results with consideration of ranging measurement error. We also provide the simulation result when the sybil node is assumed. We varies the number of neighbors (node density) from 5 to 50 with 10 – 30 trials for a verification of a given topology of neighbors. The number of phantom nodes and honest nodes are set to between 0 and 20. We assume the ranging

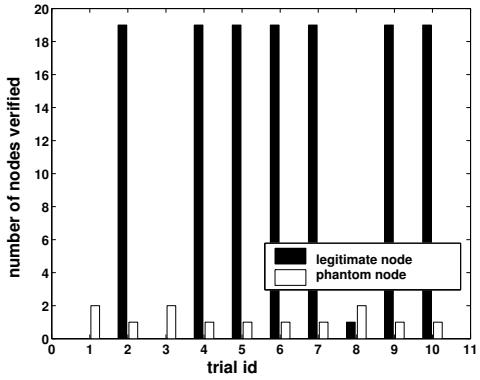

Fig. 16. Distribution of number of nodes verified (10 trials)

measurement has error rates 1%, 3% and 5%.

6.1 Case Study

We first illustrate the speculative filter through a case study. In this experiment, we speculatively choose a pair of pivots in each trial (10 trials in total), and record the number of legitimate nodes and the number of phantom nodes identified in each trial as shown in Figure 16. When 10 trials are done, with more than 99% probability, at least in one trial, two pivots are legitimate nodes. In the trial 1, 3 and 8, the consistent subsets consists of phantom nodes, but it is not selected as a final result because that the size is small compared to other trials as shown in Figure 16. The trials among 2, 4, 5, 6, 7, 9 and 10 are selected as the final largest consistent subset. In those trials, most of phantom nodes (20 phantom nodes are tried by two attackers) are filtered. A few phantom nodes is included in this example, but interestingly, these phantom nodes included are projected to the actual attacker's locations. Thus they are not actually phantom any more. Figure 17 shows the real locations of legitimate nodes (indicated by filled circles) and fake locations of phantom nodes (indicated by empty circles). The Figure 18 is the projected location from collected data. The legitimate nodes are projected to their real locations while most of phantom nodes are either filtered or projected to their actual attacker's locations.

6.2 Performance

The effectiveness of the proposed scheme is evaluated by the false positive rate, the probability that a honest node is determined as a phantom node and by the false negative rate, the probability that a phantom node is determined as a honest node, which are calculated by:

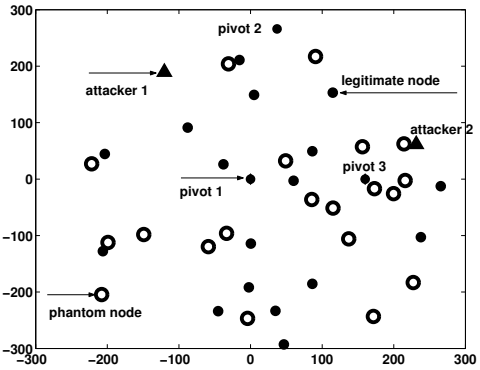

Fig. 17. Real location of honest nodes and the location attacker intended

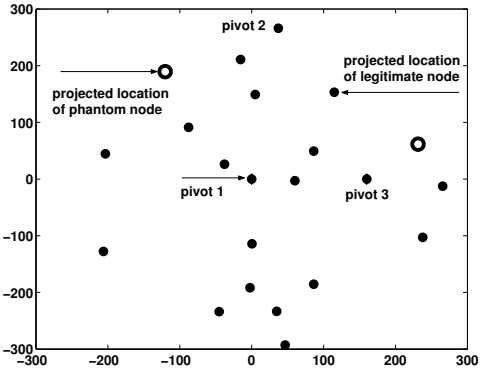

Fig. 18. Projected location of honest nodes and phantom nodes on virtual plane

$$\text{false positive rate} = \frac{\text{number of honest nodes failed verification}}{\text{total number of honest nodes}}$$

$$\text{false negative rate} = \frac{\text{number of phantom nodes authenticated}}{\text{total number of phantom nodes generated}}$$

The performance results slightly varies according to the density of neighbors, ranging measurement accuracy, and number of phantom nodes. Overall false negative rate and false positive rate is around 0.1 most of cases, showing that our solution is effective to filter phantom nodes. The false negative rate includes the case when the phantom node is projected to the attacker's actual location.

6.3 Density and Number of Trials

In the simulation we limited the number of trials so that at least one of them has all honest pivots. This can be calculated as follows: First, determine the maximum number of phantom nodes possible, which is the resiliency against the attacker. Then, from the ratio between the number of honest nodes and the maximum number of compromised nodes, we can calculate the required number of trials to ensure one of the trials has all honest pivots. When the number of honest nodes is 20 and the maximum number of phantom nodes is 20, 16 trials are required.

6.4 Localization Error

The ranging measurement error is directly related to localization error and localization error affects the performance of phantom node detection. The

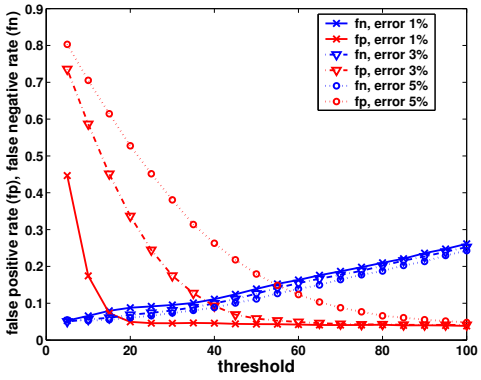

Fig. 19. Performance according to various ranging measurement error

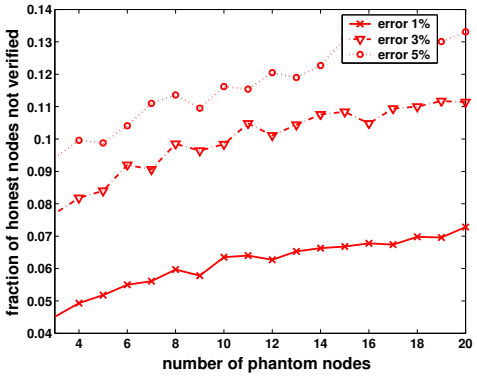

Fig. 20. False positive rate

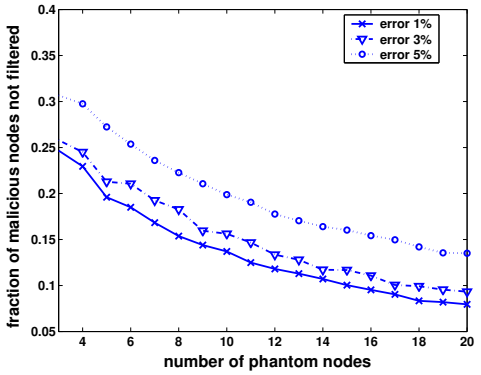

Fig. 21. False negative rate

ranging measurement accuracy depends on the media employed and the localization error depends on the localization technique used together with the ranging measurement error. We assume the displacement of the positions the attackers try to achieve is more than the random localization error.

According to our experiment, when the ranging is measured with a high accuracy, the false positive rate and false negative rate is almost zero. On the other hand, when the ranging is measured coarsely, the false positive rate and false negative rate increase. Our study in section 4.7 indicates that this is because trilateration in (32) is sensitive to the ranging measurement error, not because of our filtering solution.

Figure 19 shows the false positive rate and false negative rate according to the threshold ϵ for noise tolerance. The threshold value ϵ is the maximum acceptable difference between the distance estimate in distance measurement phase and the distance between projected nodes on the virtual plane. For a given threshold value ϵ, the false positive rate trades off with the false negative rate. When the threshold ϵ is determined more strictly the possibility that the honest nodes are not verified as legitimate increases while the possibility that the compromised nodes are filtered increases.

Figure 19 shows the performance according to various threshold values for the ranging measurement error 1%, 3%, 5%. We use the threshold value 37 for ranging measurement error 3% where the false negative rate and false positive rate are intersected. As shown in Figure 19, as the ranging measurement error increases, the false negative and false positive rate increase. While the false negative rate decreases sharply in the early stage in the graph, the false positive rate increases gradually but does not have sharp increase in the later stage. This means that the phantom node detection is not sensitive to the ranging measurement error and the value of threshold. Also, we note that the false negative rate varies in range of $0.05 - 0.25$, but the phantom nodes not filtered are projected to the actual attacker positions. This indicates that our solution can not only detect the phantom node but also possibly identify the real locations of the attackers.

6.5 Number of Phantom Nodes

We provide the simulation results when the number of phantom nodes the attacker can generate is large (more than the honest nodes). Figures 20 and 21 show the performance according to various numbers of phantom nodes. As shown in Figure 20, when the number of phantom nodes increases, the number of honest nodes excluded increase. However, the fraction of honest nodes excluded increases slightly as shown in Figure 20. Although the attacker tries to create high number of phantom nodes, the filtering procedure results in only a small number of phantom nodes attached (false negative). The interesting result shown in the Figure 21, when the total number of phantom nodes increases, the percentage of these nodes that can avoid detection reduces. Most of phantom nodes are filtered even if the number of phantom nodes increases. This is mainly because the phantom plane created by phantom nodes can only deceive the nodes that are located at the intersect line of the phantom plane and real plane as shown in Figure 4. To deceive other honest nodes, the attackers need to create a different phantom plane that intersects with the real plan at the location of individual honest node. The localized view of individual nodes enables us to filter out phantom nodes, even when the number of phantom nodes is larger than the honest nodes.

7 Related Works

We describe related works in localization technique in absence of attackers, localization in presence of attackers, location verification, and other related attacks.

7.1 Localization Techniques

In general, localization techniques can be classified largely into range-based and range-free schemes depending on whether they use ranging estimation. Range-based techniques measure physical properties such as distance or angle. Based on this information, it calculates its location. In the range-based approach the physical property is measured by Time of Arrival (ToA) (5), Received Signal Strength Indication (3; 4), Time Difference of Arrival of two different signals (TDoA) (32; 29), and Angle of Arrival (AoA) (25). Such measurement of the distance is called ranging. Range-free techniques do not measure such physical properties. As range-free approach, Bulusu et al. proposed Centroid method (6; 7), where each node estimates its location by calculating the center of the locations of all beacons it hears. Niculescu et al. (26) proposed to use the minimum hop count and the average hop size to estimate the distance between nodes. The APIT method (12) divided the environment into triangular area among beaconing nodes, and used an algorithm to calculate the maximum area, in which a node will be likely to reside. In this work, we assume we can obtain distances among the nodes based on ranging techniques (32).

7.2 Secure Localization and Location Verification

We classify the security issues in localization following taxonomy that we introduced in Section 1.1.

In the first category, *CAD*, (set of) regular nodes verify the location information of anchors, who know their location. It is equivalent to *secure localization*, that is, for uncompromised node to determine its location correctly by verifying location information of anchors. The secure localization was discussed in (17) based on directional antenna. Liu et al. (22) discussed the beacon compromise. They proposed greedy filtering and filtering by voting. Li et al. (20) introduced the way to filter beacon inconsistency by employing least median squares estimation. In those schemes, regular nodes filter locally inconsistent location informations, and determine its location based on locally consistent location information. Thus, it is essentially decentralized scheme, but the strength of security is limited to the number of anchors or their beacons in local area. For a given area, if a majority of anchors are compromised all regular nodes in the area will be deceived. In our approach, based on decentralized localization technique without purely depending on anchor node, every node takes a role of both anchor and verifier in its local map. With the proposed approach, it still be able to obtain global position by stitching local maps, or verifying anchor's claim. As both regular node and anchor (if any)

involves, the strength of security increases more than when it purely depends on anchors.

In the second category, *CPD*, few trusted, but centralized nodes validate the position or ranging claims of individual nodes. It is equivalent to *location verification*, that is, for uncompromised node to verify the neighbor's location. In *CPD*, a set of verifiers are effective to detect Sybil and Wormhole attacks independent from the number of attackers. Sastry et al. (31) dealt with location verification, where the prover is in a certain region, using propagation delay of RF signal and ultrasound. Vora et al. (38) proposed a location verification system by utilizing a topology of sensor distribution. While those works show the framework for the location verification problem, the secure localization in wireless sensor networks was first introduced in (37). In (37) they proposed to use the distance bounding protocol. For each ranging, surrounding neighbors become verifiers and they do verifiable multilateration to verify the ranging. In (8), a set of covert base stations is used for secure positioning. In (18), they proposed a way to do secure localization and location verification using anchors as verifiers. In most of these works, they assumed a set of verifiers for location verification. However, they did not provide how to protect such verifiers from adversaries. The number of verifiers is normally much less than that of regular nodes. Thus, in practice it is easy to compromise the verifiers, and once the verifiers are compromised it will have significant effects on the disruption of the verification. We overcome this issue by proposing a pure decentralized approach without the requirement of any trusted verifier.

In summary, in *CAD* the strength of security depends on anchor nodes, and in *CPD* the strength of security depends on few verifiers. Our approach, *DPD* removes the trusted centralized entities assumed in both *CAD, CPD*, and increase the strength of security. Each node plays the role of both verifier and regular node, the role of both anchor and regular node. As *DPD* does not have trusted verifiers, attackers need to compromise much more nodes to make the attack successful.

7.3 Other Existing Attacks

There are several attacks where the attacker does not directly involve in localization process, but the location related information is implicitly distorted. Sybil attack, wormhole attack, and node replication fall into this category. In these kinds of attacks, nodes have a fantasy that they are located differently from the reality. Traditionally, the Sybil Attack can be addressed by the channel bandwidth resource test and the pairwise key pre-distribution scheme proposed (24). The wormhole attack can be prevented if the propagation time of a packet is much smaller than necessary to go through the distant path

of two communicating nodes, by assuming that the location service is provided (15). In (14), the directional antenna is used to detect the inconsistency of the ranging, especially the inconsistency in the direction of the replicated message transmitted through the wormhole. Wang et al. (39) showed the centralized method to visualize the wormhole attack by only proximity of neighbors. Other than that, Parno et al. (27) proposed a distributed node-replica detection scheme where the attacker capture nodes, replicate them and insert the replicas at some locations. In our work, by obtaining each node's location correctly we can detect these kinds of attack.

8 Conclusion

Our secure localization system speculatively projects the neighboring nodes into a local map with the largest consistent subset of ranging claims. This approach authenticates the locations of honest nodes and detects the existence of the phantom nodes without relying on trusted agents. It is devised especially to be efficient when used in distributed way. Our localized construction results in quite strong results: even when the number of phantom nodes is greater than that of honest nodes, we could filter out most of the phantom nodes. In addition, our cost analysis indicates that our solution requires small overhead, and our simulation results confirm that our scheme is effective to identify the source of entities under noisy distance measurements in the presence of a large number of phantom nodes.

References

[1] Z. Abrams, A. Goel, S. Plotkin, Set K-Cover algorithms for energy efficient monitoring in wireless sensor networks, in: IPSN, 2004.

[2] K. Amouris, S. Papavassilion, M. Li, A position-based multi-zone routing protocol for wide area mobile ad-hoc networks, in: VTC, 1999.

[3] P. Bahl, V. N. Padmanabhan, RADAR: An in-building RF-based user location and tracking system, in: INFOCOM, 2000.

[4] P. Bahl, V. N. Padmanabhan, A. Balachandran, Enhancements to the radar user location and tracking system, Tech. rep., Microsoft Research Technical Report: MSR-TR-00-12 (2000).

[5] B.H.Wellenhoff, H. Lichtenegger, J. Collins, Global positions system: Theory and practice, Springer Verlag, 1997.

[6] N. Bulusu, J. Heidemann, V. Bychkovskiy, D. Estrin, Density-adaptive beacon placement algorithms for localization in ad hoc wireless networks, Tech. rep., Technical report, Computer science department, University of Southern California (2001).

[7] N. Bulusu, J. Heidemann, D. Estrin, Adaptive beacon placement, in: ICDCS, 2001.
[8] S. Capkun, M. Srivastava, M. Cagalj, Securing localization with hidden and mobile base stations, in: INFOCOM, 2006.
[9] M. Cardei, M. Thai, W. Wu, Energy-efficient target coverage in wireless sensor networks, in: INFOCOM, 2005.
[10] D. Goldenberg, P. Bihler, M. Cao, J. Fang, B. D. O. Anderson, A. S. Morse, Y. R. Yang, Localization in sparse networks using sweeps, in: MOBICOM, 2006.
[11] A. Harter, A. Hopper, P. Steggles, A. Ward, P. Webster, The anatomy of a context aware application, in: MOBICOM, 1999.
[12] T. He, C. Huang, B. Blum, J. Stankovic, T. Abdelzaher, Range-free localization schemes in large scale sensor networks, in: MOBICOM, 2003.
[13] T. He, S. Krishnamurthy, J. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, B. Krogh, An energy-efficient surveillance system using wireless sensor networks, in: MobiSys, 2004.
[14] L. Hu, D. Evans, Using directional antennas to prevent wormhole attacks, in: NDSS, 2004.
[15] Y. C. Hu, A. Perrig, D. B. Johnson, Packet leashes: A defense against wormhole attacks in wireless networks, in: INFOCOM, 2003.
[16] J. Hwang, T. He, Y. Kim, Detecting phantom nodes in wireless sensor networks, in: INFOCOM, 2007.
[17] L. Lazos, R. Poovendran, SeRLoc: Secure range-independent localization for wireless sensor networks, in: ACM WiSe, 2004.
[18] L. Lazos, R. Poovendran, S. Čapkun, ROPE: Robust position estimation in wireless sensor networks, in: IPSN, 2005.
[19] J. Li, D. S. J. Decouto, D. R. Karger, R. Morris, A scalable location service for geographic ad-hoc routing, in: MOBICOM, 2000.
[20] Z. Li, W. Trappe, Y. Zhang, B. Nath, Robust statistical methods for securing wireless localization in sensor networks, in: IPSN, 2005.
[21] D. Liu, P. Ning, Detecting malicious beacon nodes for secure location discovery in wireless sensor networks, in: ICDCS, 2005.
[22] D. Liu, P. Ning, W. Du, Attack-resistant location estimation in sensor networks, in: IPSN, 2005.
[23] D. Moore, J. Leonard, D. Rus, S. Teller, Robust distributed network localization with noisy range measurements, in: SenSys, 2004.
[24] J. Newsome, R. Shi, D. Song, A. Perrig, The sybil attack in sensor networks: Analysis and defenses, in: IPSN, 2004.
[25] D. Niculescu, B. Nath, Ad hoc positioning system (APS) using AoA, in: INFOCOM, 2003.
[26] D. Niculescu, B. Nath, DV based positioning in ad hoc networks, Telecommunication Systems.
[27] B. Parno, A. Perrig, V. Gligor, Distributed detection of node replication attacks in sensor networks, in: Security and Privacy, 2005.
[28] N. B. Priyantha, A. Chakraborty, H. Balakrishnan, The cricket location-

support system, in: MOBICOM, 2000.
[29] N. B. Priyantha, A. Miu, H. Balakrishnan, S. Teller, The cricket compass for context-aware mobile applications, in: MOBICOM, 2001.
[30] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, S. Shenker, GHT: A geographic hash table for data-centric storage in sensornets, in: WSNA, 2002.
[31] N. Sastry, U. Shankar, D. Wagner, Secure verification of location claims, in: Wise, 2003.
[32] A. Savvides, C. Han, M. Strivastava, Dynamic fine-grained localization in ad-hoc networks of sensors, in: MOBICOM, 2001.
[33] J. B. Saxe, Embeddability of weighted graphs in k-space is strongly np-hard, in: 17th Allerton Conf. Commun. Control Comput., 1979.
[34] S. S. Schiffman, M. L. Reynolds, F. W. Young, Introduction to multidimensional scaling, in: Academic Press, 1981.
[35] M. B. Srivastava, R. Muntz, M. Potkonjak, Smart kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments, in: MOBICOM, 2001.
[36] R. Szewczyk, A. Mainwaring, J. Anderson, D. Culler, An analysis of a large scale habit monitoring application, in: SenSys, 2004.
[37] S. Čapkun, J. P. Hubaux, Secure positioning of wireless devices with application to sensor networks, in: INFOCOM, 2005.
[38] A. Vora, M. Nesterenko, Secure location verification using radio broadcast, in: International Conference on Principles of Distributed Systems, 2004.
[39] W. Wang, B. Bhargava, Visualization of wormholes in sensor networks, in: WiSe, 2004.
[40] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, D. Estrin, A wireless sensor network for structural monitoring, in: SenSys, 2004.
[41] G. Zhou, T. He, S. Krishnamurthy, J. Stankovic, Impact of radio irregularity on wireless sensor networks, in: MobiSys, 2004.