Int. J. Inf. Secur. (2011) 10:155–171
DOI 10.1007/s10207-011-0127-4

REGULAR CONTRIBUTION

Secure localization with attack detection in wireless sensor networks

Wen Tao Zhu · Yang Xiang · Jianying Zhou · Robert H. Deng · Feng Bao

Published online: 19 April 2011

Abstract Rapid technological advances have enabled the development of low-cost sensor networks for various monitoring tasks, where it is important to estimate the positions of a number of regular sensor nodes whose locations cannot be known apriori. We address the problem of localizing the regular nodes with range-based location references obtained from certain anchor nodes referred to as beacons, particularly in an adverse environment where some of the beacons may be compromised. We propose an innovative modular solution featuring two lightweight modules that are for dedicated functionalities, respectively, but can also be closely integrated. First, we harness simple geometric triangular rules and an efficient voting technique to enable the attack detection module, which identifies and filters out malicious location references. We then develop a secure localization module that computes and clusters certain reference points, and the position of the concerned regular node is estimated with the centroid of the most valuable reference points identified. Extensive simulations show that our attack detection module can detect compromised beacons effectively, and the secure localization module can subsequently provide a dependable localization service in terms of bounded estimation error. The integrated system turns out to be tolerant of malicious attacks even in highly challenging scenarios.

Keywords Sensor network security · Range-based position estimation · Attack detection · Secure localization

This work was supported by the Singapore A∗STAR project SEDS-0721330047 and by the National Natural Science Foundation of China under Grant 60970138.

W. T. Zhu (✉)
State Key Laboratory of Information Security, Graduate University of Chinese Academy of Sciences, 19A Yuquan Road, Beijing 100049, China
e-mail: wtzhu@ieee.org

Y. Xiang
School of Information Technology, Deakin University, Melbourne Campus at Burwood, Melbourne, VIC 3125, Australia
e-mail: yang.xiang@deakin.edu.au

J. Zhou · F. Bao
Cryptography and Security Department, Institute for Infocomm Research, 1 Fusionopolis Way, Singapore 138632, Singapore
e-mail: jyzhou@i2r.a-star.edu.sg

F. Bao
e-mail: baofeng@i2r.a-star.edu.sg

R. H. Deng
School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902, Singapore
e-mail: robertdeng@smu.edu.sg

1 Introduction

Recent advances in wireless communications and electronics have made it possible to deploy tiny-size, low-cost, and multifunctional wireless sensor nodes, to monitor ambient environments in a certain deployment field such as temperature, pressure, and humidity, in health, home, and commercial and military areas. These sensor nodes (also referred to as sensors) are designed to monitor and report (typically to a base station) local states and events in their vicinities and a large collection of such nodes form a wireless sensor network (WSN) in a distributed and self-organizing manner [1]. The WSN has emerged as an important and economic means for monitoring the physical world and can be used for various applications like emergency response, energy management, medical monitoring, logistics and inventory management, and battlefield surveillance [2].

This work is concerned with localizing regular sensor nodes with range-based location references received from

a few specially equipped beacons nearby, whose positions can be directly obtained (e.g., via GPS). These beacons form an infrastructure and provide localization service to the rest regular nodes. A regular sensor node is usually made up of basic components including a sensing unit, a data processing unit, a transceiver unit, and a power unit. Usually, it also has a location finding system [1] (i.e., localization system [3]). The positions of sensor nodes do not necessarily need to be engineered or predetermined, which facilitates quick deployment (e.g., via random aerial scattering), particularly in inaccessible terrains or for disaster relief. Nevertheless, in many applications like target tracking, search and rescue, and wildlife monitoring, it is assumed that every regular node can discover (i.e., estimate) its actual position after deployment with its localization system. Generally, sensor nodes are supposed to be location aware so as to fulfill their monitoring tasks.

The importance of localization arises from several factors. Consider applications like building or forest monitoring, where environmental conditions must be perceived and passed on in a certain integrated manner. It does not help much if a sensor merely signals an event like "smoking", as such a signal alone is insufficient for actions to be immediately taken. Only when the detection is combined with the position of the reporting sensor node (i.e., the origin of the event) can we be clearly aware of what is exactly taking place (e.g., fire is about to start on the eastern corner of the 63rd floor). Knowledge of node positions can also assist many fundamental operations of a WSN, such as routing and key establishment. Therefore, localization plays an essential role in the development of a WSN [3,4].

WSNs are typically deployed in unattended or unsecured regions, and sensor nodes are not made tamper proof due to cost considerations. It is not unusual that an attacker targets the localization infrastructure and compromises a fraction of the beacons. Actually, sensor localization has a high risk of being subverted by malicious beacons that lie about their own positions and/or distances from the regular sensors [5]. Attacks against sensor localization can cause significant confusion and degrade the performance of other functions like routing and key establishment and thus seriously affect many WSN applications. To name a few:

- Attackers can jeopardize monitoring tasks directly dependent on sensor localization. For example, unusual heat should be reported on the summit of a mountain, but the reporting sensor node is misled into believing that it is by a river in the valley.
- Attacks against position estimation can inject so much false information that most of the routing entries kept by a sensor node become invalid, making communications between sensor nodes inefficient and consequently energy-consuming.
- Attackers may even deceive sensor nodes into believing that the network has been partitioned into geographically disjoint parts, interfering with the performance of group-oriented sensor applications.
- Attacks against location discovery can be exploited to assist other attacks to distort the network topology, e.g., to launch the sinkhole attack [6].

1.1 Related work

So far, security of location discovery in WSN has received relatively insufficient attention. Although a number of localization protocols (e.g., [4,7–9]) have been proposed [3], security has mostly been overlooked. As pinpointed in the study by [10,11], these protocols typically employ the minimum mean square error estimate (MMSEE) to gauge sensor positions by somehow solving mathematical optimization problems with certain constraints, while potential attacks from malicious adversaries are not accounted.

Recently, resilient localization in possibly adversarial settings has been recognized as an important problem, and a survey on secure localization schemes is given in the study by [12]. However, many of the schemes are based on special device assumptions. For example, HiRLoc [13] and SeRLoc [14] require extra hardware (directional antenna), SPINE [15] requires nano-second scale timing, and ROPE [16], which is a combination of SeRLoc [14] and SPINE [15], requires both extra hardware and precise time synchronization. These special requirements cannot be met on the current generation of WSNs [11].

Some more recent proposals formulate the secure localization problem as anomaly intrusion detection. These schemes are more algorithm focused (hence software oriented), but still require certain network prerequisites. For example, LAD [10] requires network deployment knowledge so that sensor nodes can verify whether the derived positions are consistent with their expectations; a failure of the verification indicates an anomaly. However, the underlying assumption may conflict with the fact that in many cases such deployment knowledge is not available apriori (actually, yet to be discovered).

Another detection-based technique by Liu et al. [17] assumes that any two communicating nodes share a unique key, while such pairwise key distribution itself is a non-trivial issue [18] (actually, sometimes in turn based on reliable localization information [19–22], resulting in a chicken-and-egg situation). This is the first issue we are concerned with; as implied in the study by [12], it is preferred that secure localization algorithms utilize non-cryptographic techniques and only rely on cryptography as the next line of defense. Second, in [17], the detection of malicious beacons is conducted by a special beacon called the *detecting node*, whose own position is assumed to be trustworthy. As we shall show later, our scheme proposed in this paper only involves attack detec-

tion by *any* regular node itself, whose location, aside from having to be accurate, even does not necessarily need to be known apriori (and is actually hidden). Third, [17] assumes that there is a wormhole detector installed on every beacon and regular node. Fourth, in [17] the replayed beacon signals are filtered out by checking the round trip time $RTT = \delta_s - \delta_r$, where δ_s and δ_r are two time differences measured by the sender (the detecting node) and the receiver (the target beacon, possibly dishonest), respectively. However, it is implicitly assumed that the target beacon, be it honest or dishonest, never manipulates its time stamps (though doing so is extremely easy) and thus does not lie on δ_r. Fifth, [17] relies on a centralized base station for the detection, while generally distributed security solutions are more preferable for WSNs.

In [23], Misra et al. identified the upper bound of the number of malicious beacons that a secure localization algorithm can tolerate and then developed an MMSEE-based algorithm known as CluRoL [5] with $\mathcal{O}(n^4 \log n)$ running time, where n is the number of available beacons (either benign or malicious). Although the time complexity seems a bit too high, Misra et al. claim that it is much lower than that of LMS [24], alleged the only other scheme with similar design goals. Another unusual attribute of CluRoL lies in that it is claimed to perform better when the malicious beacons are colluding [5], which seems counterintuitive. As we shall show later, our scheme proposed in this paper bears partial similarity to CluRoL [5] (and [25]), in that both schemes utilize certain virtual intersections. However, the processing complexity of our proposal only scales as $\mathcal{O}(n^3)$. Moreover, CluRoL assumes that the sensor nodes are preloaded with the positions of all the beacons before actual deployment, and thus, trivially, it is not viable for any beacon node to lie about its own position, which we refer to as anchor position. Again our scheme is not dependent on any such deployment knowledge apriori.

An in-depth theoretic analysis is conducted in [25] concerning algorithms that use distance information to compute locations (i.e., *range-based*). It is proved that having $\frac{n-2}{2}$ or more malicious beacons makes it impossible to localize a regular node with a bounded error, but having $\frac{n-3}{2}$ or less makes secure localization possible. This actually beats the previous upper bound $\lfloor \frac{n-4}{2} \rfloor$ identified in [23]. (The proved theorems in [25] also indicate that the secure localization problem is different from the traditional Byzantine generals problem, which at first glance bears a little similarity.) Two new algorithms are also presented in [25], the first with a worst-case computational complexity of $\mathcal{O}(n^3 \log n)$, while the second being a heuristic one without a complexity analysis. Only the simulation results for the second algorithm are presented in the study by [25], since it is the one with higher practical efficiency.

A comprehensive investigation into secure sensor localization is presented in the study by [11], where two novel methods are proposed and implemented to compare with LMS [24]. The first filters out malicious beacons on the basis of inconsistency among multiple beacon signals, while the second tolerates malicious beacons by adopting an iteratively refined voting scheme. Interestingly, in the study by [25], it is pointed out that the voting-based scheme [11] is a good example satisfying the theory developed in [25] and also that although very simple, the voting-based scheme in [11] is computationally expensive.

Recently, based on an attack-driven model specified with the Petri net, an enhanced secure localization scheme (ESLS) is proposed in [26], which extends [27] and defends against not only distance reduction attacks but also distance enlargement attacks. The major contribution lies in that, ESLS is the first to use the Petri net to validate a security scheme for WSNs. Nevertheless, ESLS relies on MMSEE for localization, as well as external mechanism such as [17] for detecting compromised beacons.

1.2 Paper organization

We observe that cryptographic techniques like authentication and encryption cannot completely prevent attacks against WSN localization; actually, many cryptographic schemes (e.g., [19–22]) themselves are location-based [18]. It is necessary to import intrusion/attack detection into sensor localization, and we pursue attack-tolerant algorithms for position estimation. Since sensors are low-cost and resource constraint, the designed algorithms should be as lightweight as possible. Particularly, we prefer the underlying rationale be as simple and intuitive as possible. In this work, we propose an efficient solution to robust localization, which features two lightweight modules (see Fig. 1): one for "triangle-based" attack detection and the other for secure localization based on "reference points". The merit of such a modular design lies in that it facilitates future upgrading of either part while not significantly affecting the overall integration.

The rest of this paper is organized as follows. Section 2 illustrates the model and terminology for WSN localization. Section 3 presents a lightweight attack detection module to shield the impact of compromised beacons. This is followed by the secure localization module presented in Sect. 4, which

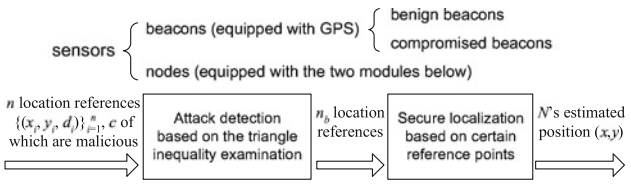

Fig. 1 Secure localization with attack detection for an arbitrary sensor N with actual deployment position (x_0, y_0), which may be any regular node in the WSN deployment field

Table 1 Quick-reference notation: symbols and parameters

B_i	A beacon with claimed anchor position (x_i, y_i)
N	Concerned regular node at unknown position (x_0, y_0)
O_i	Virtual reference circle introduced by B_i for N
$P(i, j, k)$	Reference point generated by O_i and $O_j, k = 1$ or 2
a	Offset introduced by a compromised beacon
c	Number of compromised beacons (among all n)
d_i	Distance ($< r$) from B_i physically measured by N
e_i	Physical measurement inaccuracy associated with d_i
$e_{\max}$	Max measurement error in the deployment field
e_N	Location estimation error between (x_0, y_0) and (x, y)
l	Side length of the entire sensor deployment field
n	Number of beacons available to N
n_b	Number of non-revoked beacons after the filtering
p_s	Probability for localization to turn to survival mode
r	Transmission range of a wireless sensor
$v(i, j, k)$	Number of votes received by $P(i, j, k)$
(x_i, y_i, d_i)	Location reference derived from a beacon B_i
(x, y)	Estimated value of (x_0, y_0) from location references

aims at a practical and standalone location discovery algorithm. A preliminary version of the latter module, known as Argus, appeared in [28]. Both modules involve a voting mechanism and some basic geometric principles. When employed in tandem, they can provide a very dependable and robust localization service, and this is validated with extensive simulations in Sect. 5. Finally, we conclude the paper in Sect. 6. For the sake of readers' convenience, the symbols and parameters employed throughout this paper are summarized alphabetically in Table 1.

2 Localization in sensor networks

2.1 Sensors: beacons and regular nodes

A WSN consists of wirelessly connected sensors possibly released randomly to perform various monitoring tasks. Usually, due to cost concerns, only a small percentage of the sensors, known as anchor (or beacon) nodes, in this paper *beacons* for short, are specially equipped and aware of their positions (typically via GPS). The rest, whose positions are yet to be discovered, are referred to as regular (or non-beacon) nodes, in this paper *nodes* for short (see Fig. 1).

A localization scheme (e.g., [4,7–9]) is to allow the nodes to estimate their physical locations, which essentially relies on the anchor positions that are obtained and announced by the beacons. It usually works as follows. After network deployment, the beacons broadcast radio signals. The packet carried by a beacon signal, called a beacon packet,

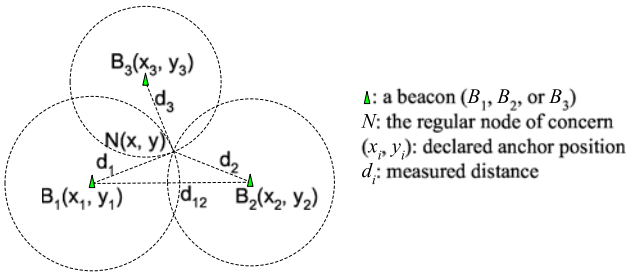

Fig. 2 Trilateration. The unknown location of a regular node N can be determined by its distances from at least three beacons, each of which declares its known position in the broadcast beacon signal. Ideally, N is positioned at the intersection of all the reference circles O_1, O_2, and O_3

encapsulates the anchor position of the beacon. Nodes not only obtain the positions claimed in the beacon packets but also can conduct certain measurements based on physical features of the beacon signals. Following the mainstream approach, we consider the range-based case [3], where the measurement is for absolute point-to-point distance between a beacon and a regular node, and can be derived from any standard methods like received signal strength indicator (RSSI), time (difference) of arrival (ToA/TDoA), and angle of arrival (AoA) [4,8].

Assume all beacons and regular nodes have the same transmission range r and communicate with each other via bidirectional wireless links. Similar to [11,25], we assume the information obtained from a beacon signal, referred to as a *location reference*, is a triplet containing the declared position (x_i, y_i) of a beacon $B_i (1 \le i \le n)$ and the distance $d_i (< r)$ locally measured by the node. When any arbitrary regular node N has collected sufficient location references (see Fig. 2), it can then estimate its deployed position (and even provide location references later to other regular nodes [3,11]).

Although some protocols like [7] adapt to the failure or addition of sensors, they generally assume all beacons are *benign* ones, while possibilities of misleading signals (e.g., carrying deceiving beacon packets) from *compromised* beacons are overlooked. In this paper, we do not differentiate between compromised, malicious, or dishonest beacons. Even if the distances (d_i's) physically measured by a regular node from the beacon signals are relatively reliable, a compromised beacon B_i can easily lie about its anchor position (x_i, y_i) as it is conveyed as digital payload information in the beacon packet to the node. Therefore, even if cryptographic technologies are employed, it is still possible for compromised beacons to declare false locations so as to deceive the nodes.

2.2 Measurement error, estimation error, and square error

In the literature [3], usually all sensors (beacons and nodes) are assumed to be statically but randomly distributed on a

two-dimension plane. Figure 2 depicts the basic principle of sensor node localization known as *trilateration*, where the position of an arbitrary point (any regular node N) in a Euclidean plane can be uniquely determined by the distances from at least three non-collinear points (nearby beacons B_1, B_2, and B_3). These beacons have known positions $\{(x_i, y_i)\}_{i=1}^{3}$ and broadcast them in the beacon signals that reach the concerned node N. To discover its location, N measures from the received signals the distance d_i to the respective beacon B_i. We call the virtual circle centered at $B_i(x_i, y_i)$ with radius d_i as a *reference circle* O_i with respect to N. Theoretically, the unknown node N is positioned at the intersection of the three reference circles O_1, O_2, and O_3.

In the real world, physically associated inaccuracy known as measurement error is intrinsic [3] (hence, no such ideal common intersection as in Fig. 2). Let (x_0, y_0) be the actual location of N. When there are no attacks, the *measurement error* of the location reference (x_i, y_i, d_i) obtained from B_i is:

$$e_i \stackrel{def}{=} \sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2} - d_i. \tag{1}$$

Usually, instead of e_i or $-e_i$, we only care about the absolute value $|e_i|$. We assume, in a given deployment field, the measurement error is bounded by $|e_i| \leq e_{\max}$, and this maximum physical inaccuracy $e_{\max}$ can be obtained experimentally.

In Fig. 2 there exist only $n = 3$ location references. Usually quite a few (e.g., a dozen) redundant location references are available, and they help improve the localization precision [11]. This more general case is called *multilateration*. The minimum mean square error estimate (MMSEE) method is to regard the location references $\{(x_i, y_i, d_i)\}_{i=1}^{n}$ as mathematical constraints that the unknown position (x_0, y_0) of the node N should satisfy, and the position is estimated by finding a solution (x, y) that satisfies the constraints but minimizes the *estimation error*:

$$e_N \stackrel{def}{=} |(x_0, y_0) - (x, y)| = \sqrt{(x_0 - x)^2 + (y_0 - y)^2}. \tag{2}$$

Usually, it is infeasible to evaluate such an estimation error, since in (2), the real (x_0, y_0) is unknown. However, analysis shows that e_N is positively correlated with the mean square error, which is practically "tangible". Given $\{(x_i, y_i, d_i)\}_{i=1}^{n}$, the mean *square error* between the calculated distances and the measured distances for the estimation result (x, y) is:

$$E(x, y) \stackrel{def}{=} \frac{1}{n} \sum_{i=1}^{n} \left(\sqrt{(x - x_i)^2 + (y - y_i)^2} - d_i \right)^2. \tag{3}$$

The minimum value of $E(x, y)$ is achieved when both its partial derivatives reach 0, and thus, the optimized (x, y) can be computed by solving a set of equations:

Table 2 A numerical example: estimated position (x, y) and the corresponding estimation error e_N when a certain (the i-th) location reference is taken as the first input for linearized MMSEE [4,11]

i	(x_i, y_i, d_i)	(x, y)	e_N
1	$(-25, -1, 27)$	$(1.59, -1.24)$	2.02
2	$(0, -15, 13)$	$(1.32, -2.00)$	2.40
3	$(15, 1, 16)$	$(0.46, -1.18)$	1.27
4	$(0, 20, 21)$	$(1.21, -0.71)$	1.40

$$\begin{cases} \frac{\partial E}{\partial x} = \frac{2}{n} \sum_{i=1}^{n} (x - x_i) \left(1 - \frac{d_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2}} \right) = 0 \\ \frac{\partial E}{\partial y} = \frac{2}{n} \sum_{i=1}^{n} (y - y_i) \left(1 - \frac{d_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2}} \right) = 0 \end{cases}. \tag{4}$$

We consider the aforementioned MMSEE approach theoretically elegant, and similar methods have been adopted in a number of localization schemes (just to name a few, [4,5,7,8,11,24,26,31]). However, the performance concern is that solving the set of equations in (4) may incur high processing overhead, which is undesirable for low-cost sensors. Although workaround like linearized MMSEE, originally proposed in [4] and then well summarized in [11], is popularly adopted to accommodate the limited computational power of current generation of sensor nodes, it inevitably increases the estimation error (2) since the square error (3) is only *approximately* minimized. Moreover, the linearization [4,11] is subject to the sequence of the inputs; our analysis and experiments have both showed that, for n available location references, there can be n *distinct* estimation results, some of which are relatively accurate but others are not so (and in practice there is no effective way to differentiate due to the unknown real (x_0, y_0)). However, so far, this uncertainty problem has not been recognized in the literature, and it makes the performances of MMSEE-based localization sometimes unstable or difficult to predict. Later, we will present a simple (yet expressive) numerical example in Table 2. On the other hand, as demonstrated in [11], the security concern for MMSEE [4] is that an attacker may introduce an *arbitrarily* large estimation error with merely a single malicious location reference. In addition, MMSEE does not provide intuitive context for attack detection. Therefore, although it is possible to build sophisticated MMSEE variants to incorporate certain security [11], we are more interested in an alternative approach for secure localization, particularly, one that eliminates estimation uncertainty.

3 Triangle-based attack detection

3.1 Assumptions

Cryptographic technologies can partially protect sensor localization. It is necessary to employ a broadcast authentication

scheme like μTESLA [2] so that regular nodes can assure the beacon packets are sent from authenticated beacons. Nevertheless, authentication alone cannot guarantee security. As sensors are not made tamper proof, an attacker may compromise a beacon to acquire cryptographic secrets to distribute authenticated but deceptive beacon packets. The attacker may also replay legitimate beacon packets, including employing wormhole attacks [6,18] where packets are tunneled from one place to another distant place to skew a node's view of available beacons. These manipulated packets bear proper authentication information and can pass cryptographic checks. To further secure sensor localization, additional mechanisms besides cryptographic means are needed. Particularly, for low-cost sensor nodes, lightweight solutions are preferred.

We assume certain broadcast authentication mechanism [2] is available, and each beacon only broadcasts one authenticated beacon signal. Hence, a compromised beacon B_i may introduce only one deceiving location reference (x_i, y_i, d_i); otherwise, it is easy for a regular node to detect the forgery by observing inconsistent behaviors of an authenticated beacon.

To introduce a misleading location reference, a malicious beacon B_i can either declare a deceptive position (x_i, y_i) in the beacon packet or manipulate the physical features of its signal (e.g., alter the radio transmission power) so that the measurement by a node results in an incorrect d_i. In the latter case, the signal should be neither too strong (otherwise such manipulation will be easily detected by a nearby sensor, be it a beacon or a regular node) nor too weak (otherwise the signal may not reach an intended node even within the range r). Therefore, we focus on the former attack, while our approach can be easily adapted. Note that an attacker may make a location reference consistent with the benign ones by manipulating both the claimed anchor position and the distance measurement; however, such a location reference will not generate significantly negative impact on location determination [11]. Also note that in replay (including wormhole) attacks, a manipulated beacon signal can be equally treated as being sent from the replaying beacon.

Following the theorems in [25] (recall Sect. 1.1), we consider (but are not limited to) the case where an attacker compromises *less than half* of the beacons available to a regular node and enough honest location references are still available. This is realistic in practical deployment: since only a small portion of the sensors are beacons, usually they are sparsely scattered so as to enable global localization. Therefore, it is unlikely for an attacker to compromise half of the beacons or more in a short period of time. In other words, it is likely for the attack to be detected before half of the beacons are compromised.

The above constraint on the attacker's ability can be loosened. For example, in case a specific set of benign beacons are given apriori, we can tolerate more compromised beacons (as long as the predetermined benign beacons are secured). Such a case is relatively easy to handle and is not our focus. In this paper, we do not depend on any predetermined benign beacons; we even allow the attacker to compromise half of the beacons or more (see Sect. 3.4), but in a slightly restrictive sense.

3.2 Attack detection framework

The goal of our attack detection is to differentiate between benign beacons and compromised beacons (recall Fig. 1). In simple words, a benign beacon is a B_i from which the measurement error (1) of the derived location reference satisfies $|e_i| < e_{\max}$, and a compromised one is a B_i with $|e_i| > e_{\max}$, where $e_{\max}$ is the maximum measurement error and is a global constant for the given deployment field [11,25]. However, comparing e_i against $e_{\max}$ is practically inapplicable. Recall that in Sect. 2.2, the estimation error (2) is intangible due to the unknown (x_0, y_0), so MMSEE has to turn to the mean square error (3). Similarly, due to the fact that (x_0, y_0) is yet to be discovered, the measurement error (1) is also intangible, and thus, there is no effective way to compare e_i against $e_{\max}$. Moreover, since sensors are error prone, occasionally it is possible for a benign beacon B_i to malfunction, resulting in an e_i slightly larger than $e_{\max}$. For compromised beacons, however, usually they try to introduce location errors significantly large ($|e_i| \gg e_{\max}$) so as to subvert the function of the WSN. To this end, a compromised B_i claims a (x_i', y_i') that is a distance away from its real anchor position (x_i, y_i). Recall from (1) that the resultant e_i is dependent on both the deceptive (x_i', y_i') and the unknown (x_0, y_0); generally, the introduced offset a has to be large enough so as to introduce a significant e_i (eventually a significant estimation error e_N).

In this work, we detect malicious attacks against localization by identifying suspicious location references, which are possibly sent from compromised beacons. Similar to [17], we employ a *detecting node*. However, in [17], the detecting node is a centralized and trustworthy beacon, essentially launching sybil attacks [18] against other beacons. In this work, to conform to the distributed nature of WSNs, we employ any regular node of concern (whose position, aside from trustworthy, is yet to be estimated) itself as a detecting node. It examines each pair of the n location references locally available to it and votes its trust on the involved beacons, and our attack detection is not in turn "attack-based"; that is, unlike in [17], the detecting node itself is not assumed to be capable of conducting sybil or other attacks.

Let N be the detecting node, which is within the communication radius r of altogether n available beacons, either benign or compromised. Recall that N is actually an arbitrary node estimating its own position. Therefore, herein, the parameter n is actually node dependent, and the attack detection framework is also in the dedicated view of N. The

proposed framework works in two stages. In the first stage, N votes *each pair* of beacons based on both location references from them, and in all, there are $\binom{n}{2} = \frac{n(n-1)}{2}$ such pairs. In most cases, n is relatively small (e.g., a dozen), and thus, the processing cost is acceptable. In the second stage, by comparing the voting result (referred to as the reputation) of *each beacon* with a certain threshold t, the detecting node identifies the (probable) compromised beacons, which essentially bear significant deviation from others.

In this method, the trust on a pair of beacons is rendered (i.e., voted) by N as a boolean: 1 for positive, 0 for negative. The decision depends on some rudimentary (thus lightweight) geometric conditions examined by N, which is detailed in the next subsection. Then, in stage two, the *reputation* of each beacon is quantitated by summing up its received votes, with $(n-1)$ being the most reputable while 0 being the most disreputable. Following previous assumptions, we generally set the detection threshold, which is the least reputation that a benign beacon is expected to achieve, to the average of the maximum and the minimum values, i.e., $t = \frac{n-1}{2}$. Exceptional thresholds are discussed in Sect. 3.4.

Note that the vote in stage one is pair oriented, whereas in stage two, the reputation is counted per beacon. In other words, one positive/negative vote simultaneously affects the reputation assessment of both involved beacons. Therefore, for any beacon, the maximum reputation $(n-1)$ may probably be achieved only when all beacons achieve the same reputation, i.e., when all beacons are benign. In case there is only one malicious beacon, it will probably receive a negative vote in each pairing, which results in the minimum reputation 0. By the end of stage two, those with a reputation lower than the threshold t are regarded compromised and will be revoked (e.g., ignored by N since then). As malicious location references are likely filtered out before estimating a node's position (to be presented in Sect. 4), the impacts of the attacks are mitigated.

Variant implementations for the above-presented attack detection are possible. For example, the detecting node N can view the n beacons as $\binom{n}{3}$ triplets, and accordingly, the vote in stage one for consistency checking is conducted on a triplet basis. In this work, we focus on pair-oriented voting for simplicity, and the according consistency examination method is proposed below.

3.3 Triangle inequality examination

In the above framework, a key issue is how to make the trust decision: in attack detection stage one, what should be the condition for a positive/negative vote? This is like a concrete "plug-in" to the generic attack detection framework. Next, we illustrate a cost-effective geometric approach termed the triangle inequality examination. Other instantiations are also possible (say, for triplet-oriented voting, but needs further investigation), which, of course, makes our attack detection module flexible and may benefit the subsequent secure localization module (recall Fig. 1).

The proposed examination is based on the triangle inequality, which is one of the elementary facts about triangles. For example, in Fig. 2, let N be the detecting node, B_1 and B_2 be the pair of beacons to be examined or voted. The three sensors determine a triangle $\triangle NB_1B_2$, with each sensor being a vertex. The *triangle inequality* tells that, for any (non-degenerate) triangle, the measure of a given side must be less than the sum of the other two sides, but greater than the difference between the two sides. For $\triangle NB_1B_2$ in Fig. 2, we should have:

$$|d_1 - d_2| < d_{12} < d_1 + d_2, \tag{5}$$

where $d_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ is calculated according to the anchor positions declared by B_1 and B_2, while d_1 and d_2 are physically measured by N from the received beacon signals. Taking the measurement errors (recall (1)) of both d_1 and d_2 into consideration, we rewrite (5) as the following criterion for actual trust decision:

$$\begin{aligned}&(\max(0, |d_1 - d_2| - 2e_{\max}))^2 \\ &\quad < (x_1 - x_2)^2 + (y_1 - y_2)^2 < (d_1 + d_2 + 2e_{\max})^2,\end{aligned} \tag{6}$$

where $e_{\max}$ is the maximum measurement error (e.g., obtained empirically) in the deployment field. Such an examination can be applied to the three dimension, as a plane can be uniquely determined by three non-collinear points (e.g., $\triangle NB_1B_2$).

Figure 2 illustrates the benign case. Now, let us assume B_2 is a dishonest beacon providing a malicious location reference (x_2', y_2', d_2), where $x_2' = x_2 + \Delta x$ and $y_2' = y_2 + \Delta y$ are deceptive coordinates declared in its beacon packet (see Fig. 3). As noted in Sect. 3.1, the location reference (x_2', y_2', d_2) may be directly from a compromised beacon B_2, or the beacon packet is originally from a genuine beacon but then replayed by the adversarial B_2. In either case, the attack can result in an estimation error that is arbitrarily large for MMSEE [11]. Next, we show how a regular node N may detect the attack, though it cannot differentiate between a directly deceiving attack or a replay attack (we shall get back to this issue in Sect. 3.5).

Concerning Fig. 3, by measuring d_1 and d_2, calculating $d_{12}' = \sqrt{(x_1 - x_2')^2 + (y_1 - y_2')^2}$, and then verifying the triangle inequality (5), it is easy for N to spot that the pair of location references (x_1, y_1, d_1) and (x_2', y_2', d_2) is conflictive. For example, in Fig. 3a, N will find that the condition $d_{12}' < d_1 + d_2$ is not satisfied, while in Fig. 3b, the condition $|d_1 - d_2| < d_{12}'$ is not satisfied. In either case, it is implied that the two reference circles O_1 and O_2 (indicated by the location references from B_1 and B_2) have no intersections, and thus, candidate locations of N do not exist (where is N?).

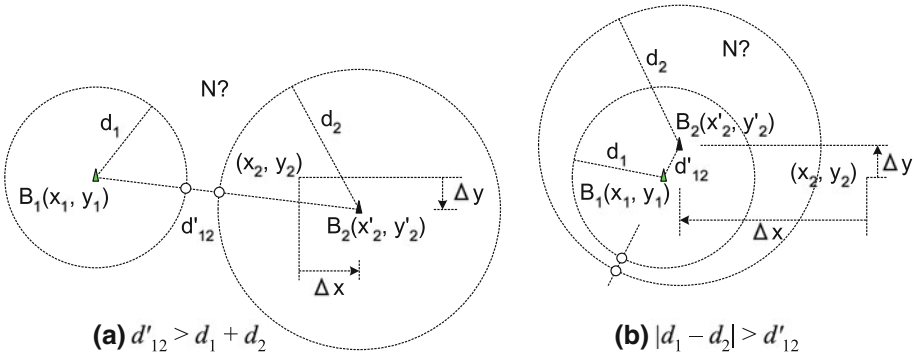

Fig. 3 A detecting node N votes a pair of beacons B_1 and B_2 according to the two location references. If the triangle inequality cannot be met, both nodes are deemed suspicious, receiving a negative vote respectively

Therefore, in attack detection stage one, N is aware of a potential attack: at least one of the two beacons is dishonest.

Given (x_1, y_1, d_1) and (x'_2, y'_2, d_2), even if B_1 is actually benign, at this stage N is not sure whether (i) only B_1 or (ii) only B_2 or (iii) both B_1 and B_2 is/are malicious. Nevertheless, as long as those pairs containing B_2 have later received enough negative votes, the final reputation of B_2 will not reach the threshold t, and thus, in stage two, N can classify B_2 as malicious. On the other hand, the benign beacon B_1, although previously suspicious, due to enough ($\geq t$) positive votes received, eventually will not be sentenced to be malicious. In a nutshell, in the attack detection framework, only beacons with high degrees of suspiciousness are considered malicious. Inevitably, there are still chances for compromised beacons to pass the triangle inequality examination, and the design goal of the lightweight attack detection module is not zero *false negative*. We put off this problem to the latter secure localization module to be presented in Sect. 4.

3.4 Rationale behind the detection threshold

To explain choices on the detection threshold, let us start with the *colluding attack*, a sophisticated deception where several malicious beacons conspire to craft fake location references that appear consistent as a whole (and hopefully receive some positive votes). There may be several forms of conspiracy. For example, the one demonstrated in [25] may break any localization scheme when $\frac{n-2}{2}$ or more compromised beacons collude. For another example, the collusion adopted in [11] is easier to understand (and to carry out), where all compromised beacons declare a bogus anchor position shifted a distance away from its true position in the same direction. We find it necessary to compare [25] with [11] and clarify that, the collusion conceived in [25] is only for a theoretical proof; the colluding attack only targets a single node deployed at a specific position (x_0, y_0), which actually is unknown (yet to be discovered). In a real WSN deployment field, however, there are a large number of regular nodes, each of which actually plays the role of a detecting node at a *hidden* position; a collusion specifically targeting a chosen (x_0, y_0) as in [25] may be easily identified by other nodes (e.g., any arbitrary N) not located at the specifically selected (x_0, y_0). On the other hand, the colluding attack presented in [11] is far more general, covert, and realistic. Therefore, later for the simulations in Sect. 5, in practice, we consider the collusion adopted in [11].

Our triangle-based attack detection, although lightweight, is (to some extent) resilient to the colluding attack. In attack detection stage one, it is possible for a pair of colluding beacons to get from the detecting node a positive vote, which favors both malicious beacons. Let c be the total number of compromised beacons, each of which may receive a positive vote in each of the $(c-1)$ pairings. However, as long as $c < \frac{n}{2}$ (recall Sect. 3.1), it is still unlikely for a malicious beacon to at least gain a reputation of $t = \frac{n-1}{2}$ in stage two (recall Sect. 3.2), which implies that eventually chances are good that most colluding nodes will be revoked. Moreover, although a benign beacon may receive a few negative votes (when paired with dishonest beacons), it will eventually receive enough positive votes so that it is not classified as malicious. In this case, we say the attack detection is a *deterministic* scheme in terms of zero *false positive*, instead of a probabilistic one where a benign beacon may sometimes be wrongly identified as compromised and thus removed.

The importance of such a deterministic property lies in that, the subsequent secure localization module (recall Fig. 1) can inherently tolerate a few compromised beacons (that have passed through the attack detection module) and perform robust position estimation, but to do so it still has to rely on enough benign beacons. Denote the probability that a beacon B_i is considered malicious as p_i. For a compromised B_i, we expect p_i to be close to 1; for a benign B_i, we expect p_i to be close to 0. Formally, a deterministic attack detection scheme is one where p_i for any benign beacon is strictly 0. This property is crucial to the secure localization module (to be presented in Sect. 4), as we depend on these benign beacons to localize the regular nodes.

Now, we consider the special case for $c \geq \frac{n}{2}$. Assume the c compromised beacons do *not* collude; otherwise, it has been strictly proved that it is impossible for any localization scheme to assure a bounded estimation error under the

scenario presented in [25]. For the non-colluding case, the assumption in Sect. 3.1 that the attacker can only compromise less than half of the beacons is actually a sufficient but not necessary condition for our attack detection module to function. This is due to the fact that even if half (or more) of the beacons are malicious, they still have to be consistent with each other so that their location reference pairs can satisfy the triangle inequality (5); otherwise, both beacons in one pairing shall receive a negative vote. Such consistency usually requires sophisticated coalition, which raises the difficulty for a successful attack. Therefore, even if half (or more) of the beacons are compromised, our attack detection module may still work. The only modification needed is that the threshold t should be reasonably lowered, so that we can still effectively tell benign beacons from compromised ones according to their reputation values computed in stage two of the framework. Since there are $(n - c)$ benign beacons, it is very likely that any one of them will get (and only get) $(n - c - 1)$ positive votes, and thus, setting $t = n - c - 1$ is the most appropriate.

3.5 Extension to collaboration

So far, the attack detection framework (instantiated with the "triangle inequality examination" plug-in) has been presented as a local process, i.e., any regular node localizing itself plays the role of a detecting node. This can be easily extended to collaborative attack detection, where multiple nodes form a *detecting group*. These nodes cooperate to share their local voting results in stage one and then deduce among the detecting group in stage two the reputation of each beacon. Such extension may enable global and more accurate detection. Another benefit is that, when a new node joins the WSN, it can be somehow provided with two lists, of known benign and compromised beacons respectively. The joining node may immediately discard beacon signals sent from known compromised beacons, and for energy conservation, it may not need to double check the known benign beacons. Moreover, if the base station is available, all local detections can be reported to the base station, which can then conduct comprehensive coordination and direction. This also helps identify replay (including wormhole) attacks (recall Sects. 3.1 and 3.3), as the base station may spot that the same beacon packet is carried by different signals (thus somewhere classified as benign but elsewhere classified as malicious).

The above extension, though favorable, may involve the following issue. Essentially, collaboration between the nodes is based on secure communication; otherwise, it may be impossible for a joining node to trust the received benign and compromised beacon lists or for a local report to be forwarded to the base station without being tampered with. Therefore, we need certain cryptographic method to protect the communication links. Many key management schemes (e.g., [19–22]) assume the deployment knowledge for efficient key establishment, while at this point, we in turn depend on cryptographic keys. Therefore, *location-based* key distribution [18] is out of the question here; we shall only employ *random* key pre-distribution techniques (e.g., [29,30]), which is less effective. Nevertheless, even if in the latter approach (e.g., [29,30]), the probability for establishing pairwise keys between neighboring nodes is significantly lower than 1, the approach may still be enough for attack detection (e.g., we can harness the birthday paradox).

Another issue that might be generally involved in any collaboration (thus not specific to collaborative attack detection for secure localization) lies in that some of the nodes in the detecting group may themselves be compromised. Possible countermeasures include applying the same concepts of voting and reputation to within the group, the elaboration of which is beyond the scope of this paper. For brevity, hereinafter, we focus on the local attack detection performed by a single detecting node N, which can actually be any arbitrary regular node. It examines the location references (with the "plug-in") and makes a judgment on its own, thus not depending on any pairwise key establishment or trust management for interaction with other nodes.

Usually, sensors are error prone (even failure prone), and sometimes, a few benign beacons that do not function well actually play the role of "less adversarial" beacons. In the next section, assuming compromised beacons has been mostly (if not all) removed, we present a practical localization scheme that is inherently tolerant of possibly malfunctioning beacons and/or those malicious beacons that have passed through the attack detection module (recall Fig. 1).

4 Secure localization based on reference points

4.1 Motivation

We have presented an attack detection module to protect WSN localization. However, that module itself is not a localization scheme. Of course, after somehow filtering out malicious beacons, a node can perform MMSEE to derive its position by solving mathematical equations (4) or so (even the attack detection can be MMSEE-based [11]). We have already designed a scheme [31] of this category, but the position estimation is still subject to uncertainty (i.e., susceptible to the sequence of inputs) since the localization in [31] employs the popularly adopted linearized MMSEE [4,11]. In this section, we alternatively present a standalone localization scheme called Argus [28], which again is based on voting (actually clustering) and employs certain geometric principles.

To explain the motivation behind this work, we consider a very simple example given in Table 2. Four location

references $\{(x_i, y_i, d_i)\}_{i=1}^{4}$, all benign, are available to a regular node deployed at $(x_0 = 0, y_0 = 0)$. We have implemented the linearized MMSEE [4,11], and the corresponding localization results are shown in Table 2: depending on which location reference is input first, linearized MMSEE provides four distinct estimation results. On the other hand, with the same set of location references, Argus [28] results in a unique estimated position $(0.76, -0.22)$ with an estimation error of 0.79, exhibiting better accuracy than all four results from MMSEE. There might also be cases when linearized MMSEE outperforms Argus (but need further investigation). Herein, we just utilize this concrete yet plain example to demonstrate the motivation for seeking an alternative solution other than MMSEE; for sensor localization, we prefer determinacy instead of uncertainty.

4.2 Reference points

The design goal of this secure localization module is to withstand a few (slightly) malfunctioning beacons and to provide a dependable position estimation service, in a robust manner when employed in conjunction with the frontal attack detection module. One observation we take advantage of is that, as the most suspicious beacons have probably been revoked by now, a reference circle O_i shall have two intersections with each of t_i other circles, where t_i is an integer close to t (the detection threshold in Sect. 3). These intersections serve as good indications for the unknown position of the concerned node (i.e., previously the detecting node, essentially any arbitrary regular node).

Suppose, n_b ($\approx n - c$) beacons have passed the attack detection. The associated n_b location references are the output by that module and are taken as the input for this secure localization module (see Fig. 1); both modules are in the local view of an arbitrary regular node N of concern. It is still possible that one or a few of these n_b beacons are actually compromised (known as *false negatives* in terms of intrusion detection). It is also possible that one or a few of the benign beacons have been wrongly considered malicious and thus removed (known as *false positives*), if the deterministic property (recall Sect. 3.4) is not assured. Therefore, for the localization module, n_b is the total number of non-revoked beacons, most (if not all) of which are benign ones supplying honest location references.

Figure 2 implies that in the ideal case where there are no measurement errors (i.e., $e_i = 0$), node N is theoretically positioned at the intersection of all the n_b reference circles ($n_b = 3$ in Fig. 2). Due to the measurement errors (recall (1)), in an actual WSN, there does not exist such a common intersection. However, we can still expect the node N to reside close to a certain set of *reference points*, which are the intersections generated by the n_b reference circles. The key idea is to treat the reference points (in all $2 \cdot \binom{n_b}{2}$ at the most) selectively, and to harness the observation that the node N should be very close to "the most valuable" reference points.

For example, in Fig. 4a where there are merely two beacons B_1 and B_2, we can only expect the node N to be located near one of the reference points N_1 and N_2, which are the intersections of reference circles O_1 and O_2. In this case, we say both points N_1 and N_2 are reliable, neither is misleading, since most compromised beacons have already been excluded. Suppose, then we receive from a third beacon B_3 the reference circle O_3, due to the addition of which N_1 and N_2 are no longer equally "valuable". With certain criterion, we can decide which of N_1 and N_2 is more valuable, as the cases illustrated in Fig. 4b and c, respectively. At the same time, the third reference circle O_3 also introduces four more reference points, among which similarly some are more valuable than others. Now that there are 3 beacons, there can be up to 6 reference points, and only a portion of them are regarded as the most valuable.

Generally, for n_b reference circles, there are $2 \cdot \binom{n_b}{2} = n_b(n_b - 1)$ reference points at the most. Now, the problem of estimating node N's position reduces to how to gauge the "value" of each reference point. Again, we employ a voting process considering certain geometric constraints. The value of a point is measured with the number of votes it receives. Those with the most number of votes form the cluster of the most "valuable" reference points. For example, in Fig. 4b, the clustering results in two points, while in Fig. 4c the result is three points.

For brevity, we omit the rare tangent case that two reference circles have exactly one intersection. Thus, a reference circle O_i has either zero or two intersections with any other circle O_j, for $1 \le i, j \le n_b, i \ne j$. If they have two intersections, we denote the one with a less value in the horizontal axis (x-coordinate) as $P(i, j, 1)$, the other as $P(i, j, 2)$. Note $P(i, j, k) = P(j, i, k)$ for any $i \ne j$ and $k = 1, 2$. To

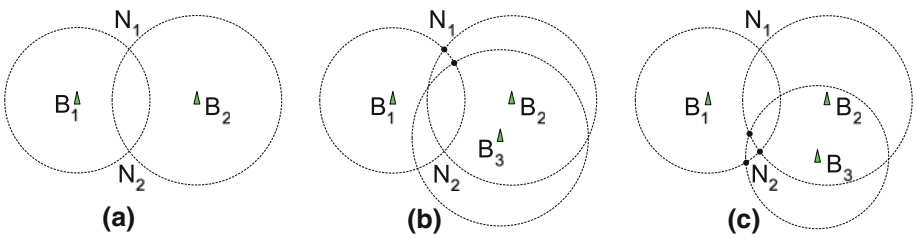

Fig. 4 Intersections of reference circles serve as good indications of the node's position to be discovered. It is expected to be very close to the most valuable reference points, where there are two in case (**b**) or three in case (**c**)

gauge the value of a reference point $P(i, j, k)$, $1 \le i < j \le n_b, k = 1, 2$, we associate it with a voting counter $v(i, j, k)$, initialized to 0 and incremented by 1 each time this $P(i, j, k)$ receives a positive vote.

4.3 Voting/clustering the reference points

Our method for voting the reference points only involves basic arithmetic operations. To begin with, we define a function d^2 for any two points $A_1(x_1, y_1)$ and $A_2(x_2, y_2)$:

$$d^2(A_1, A_2) \stackrel{def}{=} (x_1 - x_2)(x_1 - x_2) + (y_1 - y_2)(y_1 - y_2). \tag{7}$$

It is the square distance between the two points and can be easily implemented in any low-cost processor that supports float-point addition and multiplication. Since it will be frequently invoked by the secure localization module (also the attack detection module; recall (6)) as a basic primitive, it should be as lightweight as possible.

Similarly to the attack detection module, the secure localization module works in two stages. In stage one, we have each reference circle "vote" on the reference points. As one may expect, for any reference point $P(i, j, k)$, we do not count the votes from circles O_i and O_j themselves, as any beacon $B_l(x_l, y_l)$ always trusts all the points positioned on its reference circle O_l.

Given a reference point $P(i, j, k)$, for any beacon B_l with location reference (x_l, y_l, d_l), $1 \le l \le n_b, l \ne i, l \ne j$, the node N considers the distance d between $P(i, j, k)$ and B_l. If d is between $\max(0, d_l - e_{\max})$ and $(d_l + e_{\max})$, we say $P(i, j, k)$ complies with the location reference from B_l and increase the voting counter $v(i, j, k)$ by 1. (In Fig. 4, for simplicity, the maximum measurement error $e_{\max}$ is not rendered.) Similar to (6), the real-world condition for actual examination is as follows:

$$(\max(0, d_l - e_{\max}))^2 < d^2(P(i, j, k), B_l) < (d_l + e_{\max})^2. \tag{8}$$

For $1 \le i < j \le n_b$ and $k = 1, 2$, there can be up to $n_b(n_b - 1)$ reference points. As each $P(i, j, k)$ is associated with a $v(i, j, k)$, the node N estimating its position needs to maintain $n_b(n_b - 1)$ voting counters at the most, each of which can be implemented with just one byte. Each $P(i, j, k)$ receives from B_l, $1 \le l \le n_b, l \ne i, l \ne j$ in all $(n_b - 2)$ votes, either positive or negative. So, condition (8) is totally checked $n_b(n_b - 1)(n_b - 2)$ times at the most. That is, the primitive d^2 in (7) is invoked $n_b(n_b - 1)(n_b - 2)$ times at the most. Note that $n_b \approx n - c$ is relatively small, where n is the number of beacons whose radio signals can reach N. Usually, a regular node N only needs to estimate its position once [17]. Hence, the storage and processing costs for N are still acceptable.

We summarize the voting algorithm presented above with C-like code in Algorithm 1, where the keyword `continue` is utilized to skip the current execution within a loop. Unlike Sect. 3.5, this module incurs no communication cost at all. After the voting, the most "valuable" reference points are clustered by singling out the points that have received the most votes. We then choose the geometric centroid of these points as the estimated position (x, y) of N actually residing at unknown (x_0, y_0).

Algorithm 1: Voting the reference points for clustering

```
for (i=1; i<=n-1; i++) {
   for (j=i+1; j<=n; j++) {
       if (intersecting(O_i, O_j) == false) continue;
       ... //two intersections P(i,j,1) and P(i,j,2)
       for (k=1;k<=2;k++) {
           v(i,j,k) = 0;
           for (l=1; l<=n; l++) {
               if (l==i||l==j) continue;
               D=d^2(P(i,j,k),B_l);
               if (max(0,d_l-e_max)*max(0,d_l-e_max) <D
               && D<(d_l+e_max)*(d_l+e_max)) v(i,j, k)++;
           }//l
       }//k
   }//j
}//i
```

4.4 Further discussions on intersections

The voting process depicted in Fig. 4 and Algorithm 1 can yield a cluster of the most valuable reference points that are very close to each other (and very close to the concerned regular node N). The geometric centroid (x, y) of the cluster can be easily computed by calculating the arithmetic mean with each coordinate of the points. For example, if only two points are left (as in Fig. 4b), the centroid will be the middle point of them; if three points are left (as in Fig. 4c), the centroid will be the intersection of the three medians of the triangle determined by them, i.e., the triangle's barycenter.

Nevertheless, at this final step, instead of offering N's estimated position, our algorithm may occasionally fail. This is attributed to the rare case that no reference points are available at all. Recall that in the triangle inequality examination, from formula (5) to its rewritten form (6), we have considered $e_{\max}$ for both d_1 and d_2, and thus, the criterion for trust decision allows the total error pertaining to measurement inaccuracy to be $2e_{\max}$. However, for two reference circles O_1 and O_2 to have intersections (i.e., to generate reference points), the triangle inequality (5) has to be strictly met (recall Fig. 3). Clearly, (6) is much looser than (5). This explains why it is possible (though rare) for the n_b reference circles that have passed the attack detection to turn out to have no intersections.

Then, as a workaround, our algorithm switches to a so-called *survival* mode: we connect the two centers of each

pair of the n_b reference circles and specify the two (of four) adjacent intersections of the line with the two circles as reference points. We reuse Fig. 3 to demonstrate the survival mode, where the specified reference points are marked as hollow dots. As one may expect, for the survival mode, we do not vote/cluster these $2 \cdot \binom{n_b}{2} = n_b(n_b - 1)$ intersections; instead, we directly compute their centroid as the discovered position (x, y) for N. Let p_s be the probability that the localization turns to the survival mode. According to the above analysis, p_s only becomes detectable when n_b is quite small: the fewer reference circles available, the more possible they have no intersections (this implicitly explains the reason that we cannot rely on Argus itself for the attack detection). Nevertheless, in our extensive simulations, we find out $p_s < 1\%$, even if n_b is as few as 4.

Generally, the running time of the attack detection module is $\mathcal{O}(n^2)$ (recall Sect. 3.2) and that of the secure localization module is $\mathcal{O}(n^3)$. Therefore, the overall processing cost still scales as $\mathcal{O}(n^3)$. Recall that a reference point is an intersection of two reference circles (except for our survival mode). Existent schemes utilizing similar intersections are found in [5] and [25] (recall Sect. 1.1), whereas the former scheme known as CluRoL incurs a processing cost of $\mathcal{O}(n^4 \log n)$ and the latter incurs $\mathcal{O}(n^3 \log n)$.

5 Simulation results

5.1 Preliminaries

Next, we present the simulation results for our attack detection (based on the triangle inequality examination) in conjunction with our secure localization (based on reference points) to validate the overall modular design shown in Fig. 1. The evaluation focuses on the functionality of the two modules under various conditions. In all simulations, without loss of generality, the target field of interest is instantiated in the view of (an arbitrary) regular node N, modeled as a circle centered at N with radius r. Let there be n beacons $\{B_i\}_{i=1}^{n}$ randomly deployed within the circle so that their beacon signals can reach N. For convenience, we set the origin of the coordinates at the deployment point of N (hence $x_0 = y_0 = 0$). We model the WSN deployment field as an $l \times l$ square field centered at N, with the side length l much larger than r. As the target field of interest should only be a very limited area of the entire deployment field, we assume n (in the view of node N) is relatively small. Generally, we set, $n = 11$, $r = 50\,\text{m}$, and $l = 600\,\text{m}$ ("m" is just a relative unit of distance). One can turn to Table 1 for a quick reference of the notation.

Let the first c of the n beacons be compromised ones and the rest $(n - c)$ be benign ones. For example, Fig. 5 depicts a target field with $n = 11$ beacons, $c = 4$ compromised (in black), and $n - c = 7$ benign (in green). In our simulations,

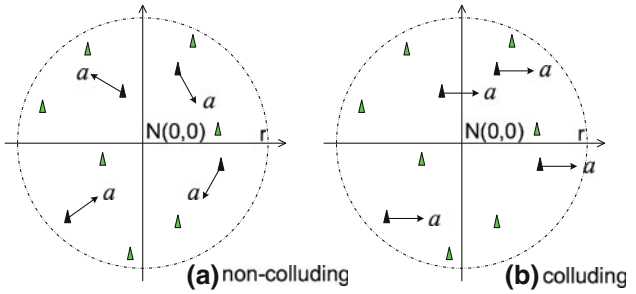

Fig. 5 In the view of a node N, the target field is within a circle of radius r centered at itself, containing n randomly deployed beacons. In attack scenario (**a**), each of the c compromised beacons declares a position a meters away from its real anchor position in an arbitrary direction. In attack scenario (**b**), each compromised beacon increases its horizontal coordinate by $\Delta x = a$ meters

the anchor position (x_i, y_i), be B_i benign or compromised, is generated as follows (i begins with 1). Select $x_i, y_i \in_R [-\frac{l}{2}, \frac{l}{2}]$ and check whether the distance between $N(0, 0)$ and the candidate $B_i(x_i, y_i)$ is within r. If not, discard the candidate and repeat the above select-check process; otherwise, record B_i and yield B_{i+1} similarly (till B_n). Then, following the measurement error defined in (1), we model the distance measured from B_i as $d_i = \sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2} - e_i = \sqrt{x_i^2 + y_i^2} - e_i$, where similar to [11], the uniform distribution $e_i \in_R [-e_{\max}, e_{\max}]$ is assumed for the physical inaccuracy. Compared with the normal distribution employed in [31], the uniform distribution actually implies larger measurement errors in general and thus is more challenging to sensor localization.

We refer to the randomly deployed B_i, $1 \le i \le n$ as a beacon *layout*. To illustrate different attack scenarios (explained later), Fig. 5a and b employ the same layout, which is not necessary the case in the simulations. In fact, the two proposed modules are simulated with C program on a Linux platform, with each beacon layout randomized by a seed obtained via the system call `gettimeofday` claiming microsecond-level timing; all beacon layouts in our simulations ($\sim 10^6$ rounds) are different.

For each combination of the parameters (e.g., the number of compromised beacons c, the maximum measurement error $e_{\max}$), we run the simulation 10^4 rounds (10^3 or so is not enough to observe the survival mode). Then, we draw statistics on p_i (recall Sect. 3.4), the detection rate for a compromised $B_i (1 \le i \le c)$ or the false positive rate for a benign $B_i (c + 1 \le i \le n)$. Let r_c be the number of revoked compromised beacons (which averages $\sum_{i=1}^{c} p_i \approx c$) and r_b be the number of removed benign beacons (which averages $\sum_{i=c+1}^{n} p_i \approx 0$). In each round, we redirect the residual $n_b = n - r_c - r_b$ location references output by the attack detection module as input to the secure localization module.

5.2 Triangle-based attack detection

To evaluate the attack detection based on the triangle inequality examination, following the discussions in Sect. 3.4, we consider two different attack scenarios. In the first scenario, the c compromised beacons are non-colluding. Each forges a position a meters away from its real location, in an unorganized random manner (see Fig. 5a). This is modeled as $\Delta x = a\cos\varphi$ and $\Delta y = a\sin\varphi$ (recall Fig. 3), where $a > 0$ and $\varphi \in_R [0, 2\pi)$. The second scenario is a colluding one borrowed from [11]. As depicted in Fig. 5b, the c compromised beacons conspire to declare deceiving locations that appear consistent themselves, setting $\Delta x = a$ meters away ($\Delta y = 0$) from their actual anchor position, respectively. That is, each increases its horizontal coordinate by a meters, in an organized and unified manner ($\varphi = 0$), trying to create a virtual location in the horizontal axis that is a meters away from N's real position.

In either of the two scenarios, it tends to be less beneficial for the adversary to launch attacks with a deviation a only on the order of $e_{\max}$ (recall Sect. 3.2); this way the dishonest location references just serve more or less like honest ones. Therefore, we designate a to be on the order of r (but not necessarily of l), so that the compromised beacons may effectively mislead the node N. Particularly, we consider cases for $r \le a \le 4r$. Since it is impossible for the detection rate $p_i (1 \le i \le c)$ to be always 1, in some rounds, it is possible for $n_b > n - c$, i.e., the residual n_b nodes contain compromised beacons that happen to have passed the attack detection.

For some reason, we first present the simulation results regarding the second attack scenario (colluding, $\varphi = 0$). In this case, following Sect. 3.2, we plainly set $t = \frac{n-1}{2}$ and consider $c < \frac{n}{2}$, specifically, $t = 5$ and $c = 2, 3, 4, 5$. We vary $e_{\max}$ from $4\%r$ to $16\%r$ (which is comparable to the choices in [25]), particularly, $e_{\max} = 2.0, 4.0, 6.0, 8.0$. For each combination of $(a, c, e_{\max})$, we run the simulation 10^4 times to draw statistics of p_i, $1 \le i \le n$ and thus r_c and r_b (recall Sect. 5.1). The results show that, while the detection rates p_i, $1 \le i \le c$ are not always close to 1, the false positive rates p_i, $c + 1 \le i \le n$ are always exactly 0. This conforms to the deterministic property (recall Sect. 3.4) that a benign beacon will never be mistaken for a malicious one. Consequently, r_b is always 0, while r_c is not always close to c, as depicted in Fig. 6.

Figure 6 reveals that the attack detection module is insensitive to $e_{\max}$, as the 4 patterns illustrated under different $e_{\max}$'s are quite similar. Although a larger $e_{\max}$ may result in a slightly smaller r_c under the same (a, c), such an impact is marginal. Therefore, later, we fix $e_{\max} = 10\%r = 5.0$ in simulating the non-colluding scenario. Another observation is that r_c increases with a, but when $a > 3.0r$, r_c becomes very close to c. This can be explained as follows. When a is relatively small, the localization error introduced by the compromised beacons tends to be limited; these beacons may look like benign ones and thus are not removed. Recall that our triangle inequality examination is just a lightweight "plug-in" for low-cost sensor nodes to identify only the most suspicious beacons. When a is observably large, however, almost all c compromised beacons are removed, implying that the malicious attacks are detected and restrained very effectively.

For the non-colluding attack scenario, we set $e_{\max} = 5.0$ as afore-discussed, and we allow $c \ge \frac{n}{2}$, particularly, $c = 4, 5, 6, 7$. Recall Sect. 3.4 that $c < \frac{n}{2}$ is a sufficient but not necessary condition for our attack detection to function. In this case, we need to lower the threshold from $t = \frac{n-1}{2}$, so that we can still separate benign beacons from compromised ones based on their reputations in attack detection stage two. Particularly, we check four thresholds $t = 2, 3, 4, 5$. Again, let the introduced offset be $r \le a \le 4r$. For each combination of (a, c, t), we run the simulation 10^4 times to draw statistics of p_i, $1 \le i \le n$ and thus r_c and r_b. The results show that, while r_c tends to converge to c (as we can expect from Fig. 6), r_b is not always 0, not achieving the deterministic property.

As depicted in Fig. 7, r_c increases with a in a similar manner as in Fig. 6; when $a > 3.0r$, r_c becomes very close to c, except for $t = 2$ as in Fig. 7a. This exception can be attributed to that $t = 2$ significantly lowers the bar for accepting a

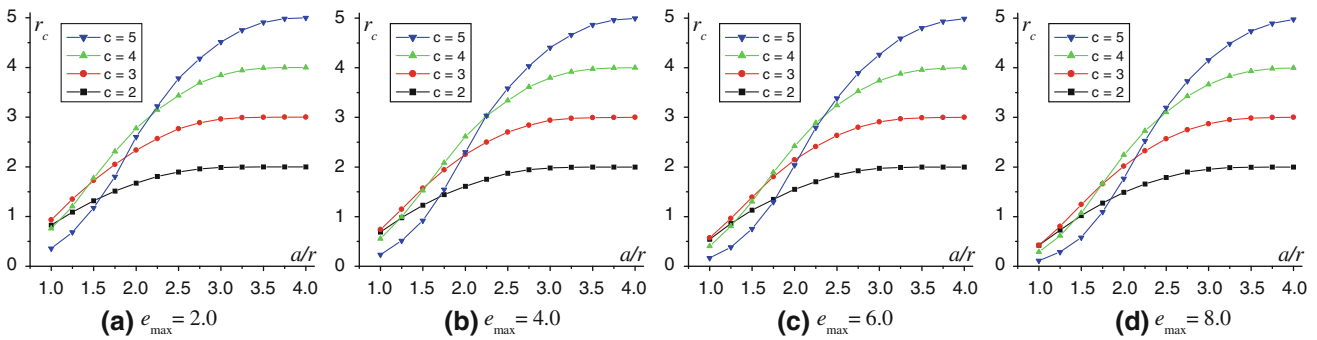

Fig. 6 Number of removed compromised beacons r_c under different $(a, c, e_{\max})$ in the colluding scenario, where a is the offset introduced by each of the c compromised beacons, and $e_{\max}$ is the maximum measurement error

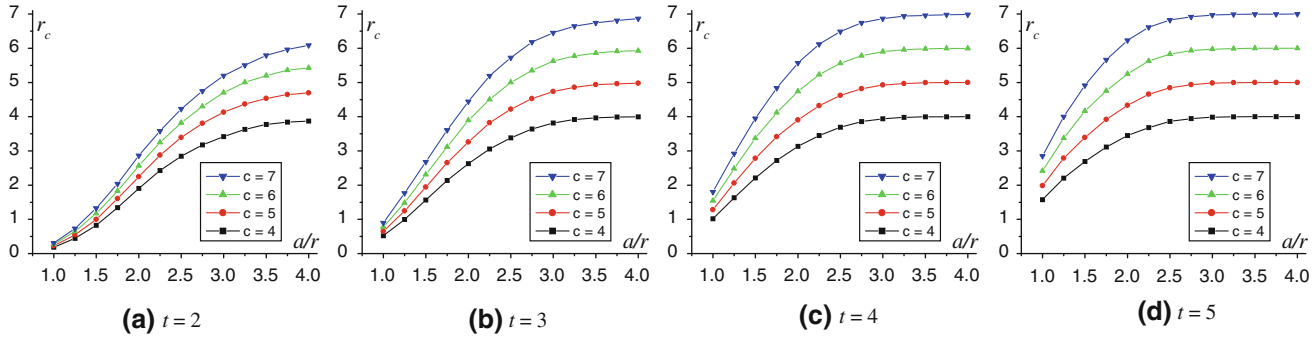

Fig. 7 Number of removed compromised beacons r_c under different (a, c, t) in the non-colluding scenario ($e_{\max} = 5.0$)

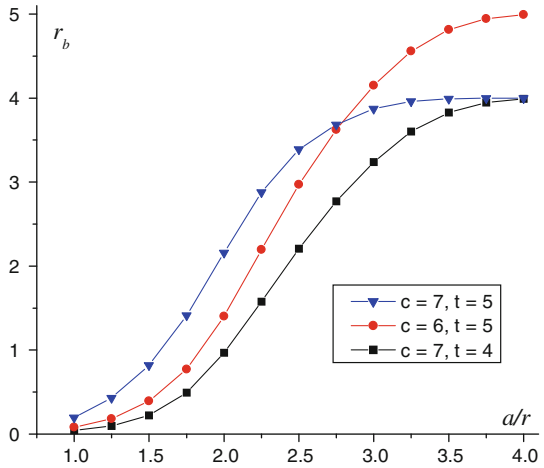

Fig. 8 Number of removed benign beacons r_b under different (a, c, t) in the non-colluding scenario ($e_{\max} = 5.0$)

benign beacon, as one needs mere two positive votes to pass the attack detection. At the same time, this can assure the deterministic property, as with a lower threshold ($t = 2, 3$), we find r_b to be always 0. In Fig. 7, for a higher threshold, ($t = 4, 5$), r_c converges more quickly to c, but the price is that in the three combinations depicted in Fig. 8, r_b also converges to $(n - c)$. Note that it is likely for a benign beacon to only receive a positive vote when paired with another benign beacon, but there are only $(n - c - 1)$ such pairs. Thus, when $t > n - c - 1$ (i.e., $t + c \geq 11$, as with all three combinations in Fig. 8), even benign beacons cannot obtain a reputation of at least t. Therefore, r_b converges to $(n - c)$, which implies that not only the c compromised but also the $(n - c)$ benign beacons are removed by the attack detection module.

On the one hand, from Fig. 8, we know a threshold as high as $t = 5$ breaks the deterministic property and results in that all the beacons are removed, completely disabling the secure localization following the attack detection. On the other hand, from Fig. 7, we know that a threshold as low as $t = 2$ degrades the detection performance. Therefore, when $c \geq \frac{n}{2} = 5.5$ in the non-colluding scenario, we choose $t = 4$ for $c = 6$ and $t = 3$ for $c = 7$. Note that these combinations of (t, c) exactly conform to our analysis in Sect. 3.4, where $t = n - c - 1$ is supposed to be the most appropriate threshold.

Of course, in a real WSN, it may be difficult to figure out this question: if there is no colluding, among the total n beacons in the target field, how many of them are compromised? (i.e., $c =?$) As a result, in the non-colluding scenario, although it is allowed $c \geq \frac{n}{2}$, we still have to assume c is somehow bounded so that we can consider an appropriate detection threshold t, just like in the colluding scenario (where we follow the strict assumption that $c < \frac{n}{2}$ so that we can designate $t = \frac{n-1}{2}$).

5.3 Localization based on reference points

After revoking the identified malicious beacons, the attack detection module passes the residual n_b location references (recall Fig. 1) to the secure localization module for system integration. Due to false negatives (Figs. 6, 7) and false positives (Fig. 8), the n_b nodes may not be exactly the $(n - c)$ benign beacons. For the concerned node N which actually resides at the origin of the coordinates, following (2), we investigate the estimation error $e_N = \sqrt{x^2 + y^2}$, which indicates how far away the estimated position (x, y), is from N's real deployment location (x_0, y_0), in an average sense from the 10^4 rounds of simulations with a certain parameter combination. We expect the integrated system to be (i) *dependable* in that e_N is only on the order of the maximum measurement error $e_{\max}$ and (ii) *robust* in that the system still works even if a majority of the beacons are compromised (e.g., $\frac{c}{n} \approx \frac{2}{3}$).

Again, we first consider the colluding scenario, where $t = \frac{n-1}{2}$ since $c < \frac{n}{2}$. We plot the average position estimation error e_N under various conditions in Fig. 9. It shows that e_N is more or less proportional to $e_{\max}$. In most cases, e_N is even less than $e_{\max}$. The only exception is that when there are 5 compromised beacons (which is the largest c the algorithm can tolerate) and a is relative small ($a \leq 2.0r$), e_N can be significantly large. This can be understood with respect to

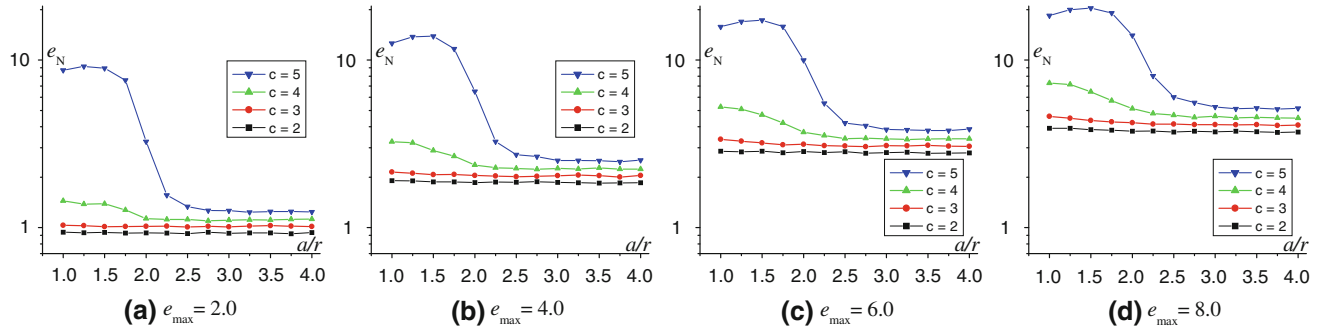

Fig. 9 Average estimation error e_N, estimating (x_0, y_0) with (x, y) under different $(a, c, e_{\max})$ in the colluding scenario

Fig. 6, as when $c = 5$ and $a \leq 2.0r$, the number of removed compromised beacons r_c is unusually low (e.g., even lower than the case when $c = 3$, as shown in Fig. 6). Consequently, we attribute the large estimation errors to the false negatives. The largest e_N appears to be 20.428, occurring at ($a = 1.5r = 75$, $c = 5$, $e_{\max} = 8.0$) in Fig. 9d. Nonetheless, even so the ratio of e_N to $e_{\max}$ is only 2.55, and the ratio of e_N to a is as low as 0.27. Note that in Fig. 6, when $a > 3.0r$, r_c becomes very close to c, i.e., almost all compromised beacons are removed (while $r_b = 0$, i.e., no benign ones are revoked). This is resonated in Fig. 9 with e_N being almost constant (only associated with $e_{\max}$ and c but not with a). When $a > 3.0r$, the malicious location references can no more affect the secure localization module. Hence, we argue that our secure localization is dependable, even when there are extremely deceptive location references ($a \gg e_{\max}$).

As to the non-colluding scenario, an important fact is that for MMSEE-based sensor localization, even a single compromised beacon can introduce an unlimited estimation error; specifically, e_N may grow almost directly proportional to the introduced deviation a [11]. For our secure localization module, since the non-colluding compromised beacons are unorganized (recall Fig. 5a) and thus less vicious, one can expect that under comparable circumstances, it should perform better than with colluding beacons. Therefore, we only consider challenging cases where $c \geq \frac{n}{2}$, i.e., we are interested in how the whole system responds to highly adversarial environments when more than half of the available beacons are compromised.

Note for any localization scheme to function, at least three location references are needed (as in Fig. 2). Moreover, Fig. 8 warns that it is possible for not only the compromised but also the benign beacons to be removed by the attack detection module, which may result in $n_b = 0$ hence completely disabling the secure localization module. Therefore, following the previous subsection, we only consider two combinations ($c = 6, t = 4$) and ($c = 7, t = 3$). Note that $c = 7$ is a very challenging case for the two modules to handle, as there are only $(n - c) = 4$ benign beacons to depend on. In this case, as aforementioned in Sect. 4.4, the survival mode is turned to with a probability $p_s \approx 0.2\%$.

The simulation results for ($c = 6, t = 4$) and ($c = 7, t = 3$) are shown in Fig. 10. Similarly to the colluding case depicted in Fig. 9, e_N scales with $e_{\max}$, but interestingly, larger a's usually result in smaller e_N's, ending up with bounded impacts on the position estimation. When there are $c = 6$ compromised beacons, from Fig. 7c, we know that r_c is larger than 5 when $a > 2.0r$. Accordingly, in Fig. 10a, e_N favorably drops below $e_{\max}$ when $a > 2.0r$. However, this phenomenon is not observed in Fig. 10b, where larger

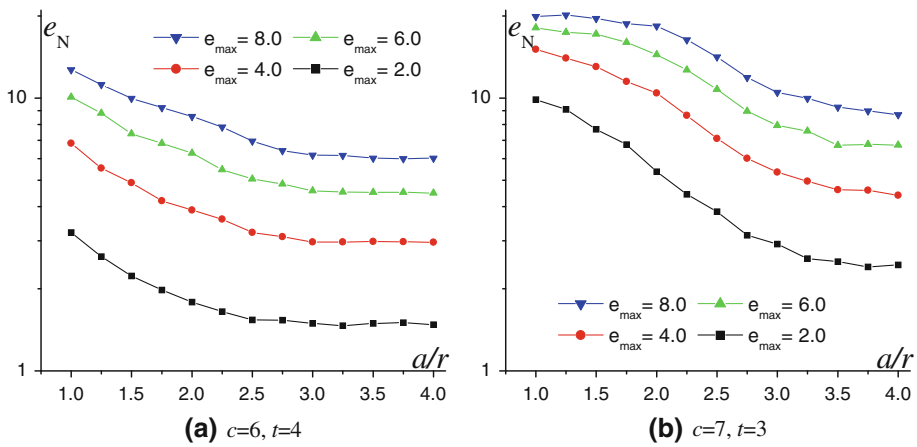

Fig. 10 Average location estimation error e_N for $c \geq \frac{n}{2}$ under different $(a, e_{\max})$ in the non-colluding scenario

e_N's are incurred due to the only availability of $n - c = 4$ honest location references. Actually, when there are $c = 7$ compromised beacons, they occupy $\frac{c}{n} \approx 64\%$ of all the beacons. This tends to be extremely hostile, but our system still turns out to be resilient to the attack as shown in Fig. 7b. The largest e_N appears to be 20.156, which occurs at ($a = 1.25, r = 62.5, e_{\max} = 8.0$) in Fig. 10b. Nonetheless, even so the ratio of e_N to $e_{\max}$ is only 2.52, and the ratio of e_N to a is as low as 0.32. Therefore, the whole system proves to be dependable and robust.

6 Conclusion

In many sensor network applications, it is a fundamental and important task to localize the deployed sensors, most of which are just low-cost regular nodes, with a certain location finding system. It is preferred that such a system be self-organized and attack tolerant, with the only assistance from a limited number (n) of special nodes called beacons, whose positions are readily available right after the deployment but some of which may possibly be compromised. If the location finding system is without protection, an attacker may easily mislead the position estimation by regular nodes and even subvert the normal operation of sensor networks.

In this paper, we addressed the problem of secure sensor localization, in the presence of both benign and compromised beacons, with a novel modular solution. The proposal features two lightweight modules, which are for dedicated functionalities respectively but can also be closely integrated. Accordingly, our technical contributions are twofold. First, based on the geometric triangle inequality, we developed a lightweight attack detection module employing certain voting mechanism. Due to its simplicity in attack detection and mitigation, it is a kind of defense preferable for low-cost sensors, and simulation results have also validated its effectiveness. Second, we developed a standalone secure localization module that can intrinsically tolerate some malfunctioning beacons. By voting and clustering certain reference points, it can provide practical and efficient location discovery. Extensive simulations show that when employed in tandem with the attack detection module, the secure localization module can provide a dependable and robust position estimation service even in highly challenging conditions.

Due to the modular design, the developed solution is flexible and extensible in nature, which facilitates future improvement. Moreover, our proposal does not depend on special device assumptions (like extra wireless hardware or precise time synchronization) that are inapplicable to the current generation of WSNs, nor is it based on sensor nodes with special knowledge (like deployment information known apriori) or unusual capabilities (like being able to conduct probes based on the sybil attack). Furthermore, the overall running time for the integrated system only scales as $\mathcal{O}(n^3)$, which is lower than that of comparable schemes ($\mathcal{O}(n^4 \log n)$ for [5] and $\mathcal{O}(n^3 \log n)$ for [25]) and thus is more favorable for resource-constrained sensor nodes.

Acknowledgments We thank anonymous reviewers for checking carefully the manuscript and raising valuable comments.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Commun. Mag. **40**, 102–114 (2002)
2. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: SPINS: Security protocols for sensor networks. Wirel. Netw. **8**, 521–534 (2002)
3. Boukerche, A., Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F.: Localization systems for wireless sensor networks. IEEE Wirel. Commun. **14**, 6–12 (2007)
4. Savvides, A., Han, C.-C., Strivastava, M.B.: Dynamic fine-grained localization in ad-hoc networks of sensors. In: Proceedings of the 7th International Conference on Mobile Computing and Networking (MobiCom'01), pp. 166–179 (2001)
5. Misra, S., Xue, G.: CluRoL: Clustering based robust localization in wireless sensor networks. In: Proceedings of the IEEE Military Communications Conference 2007 (MILCOM'07), pp. 1–7 (2007)
6. Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: attacks and countermeasures. Ad Hoc Netw. **1**, 293–315 (2003)
7. Nagpal, R., Shrobe, H., Bachrach, J.: Organizing a global coordinate system from local information on an ad hoc sensor network. In: Proceedings of the 2nd International Workshop on Information Processing in Sensor Network (IPSN'03) Lecture Notes in Computer Science, vol. 2634, pp. 333–348 (2003)
8. Niculescu, D., Nath, B.: Ad hoc positioning system (APS) using AoA. In: Proceedings of the 22nd Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03), vol. 3, pp. 1734–1743 (2003)
9. Thaeler, A., Ding, M., Cheng, X.: iTPS: an improved location discovery scheme for sensor networks with long-range beaconsc. J. Parallel Distrib. Comput. **65**, 98–106 (2005)
10. Du, W., Fang, L., Ning, P.: LAD: Localization anomaly detection for wireless sensor networks. J. Parallel Distrib. Comput. **66**, 874–886 (2006)
11. Liu, D., Ning, P., Liu, A., Wang, C., Du, W.K.: Attack-resistant location estimation in sensor networks. ACM Trans Inf Syst Secur **11**, 22:1–22:39 (2008)
12. Boukerche, A., Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F.: Secure localization algorithms for wireless sensor networks. IEEE Commun. Mag. **46**, 96–101 (2008)
13. Lazos, L., Poovendran, R.: HiRLoc: high-resolution robust localization for wireless sensor networks. IEEE J. Sel. Areas Commun. **24**, 233–246 (2006)
14. Lazos, L., Poovendran, R.: SeRLoc: robust localization for wireless sensor networks. ACM Trans. Sens. Netw. **1**, 73–100 (2005)
15. Čapkun, S., Hubaux, J.-P.: Secure positioning of wireless devices with application to sensor networks. In: Proceedings of the 24nd Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05), vol. 3, pp. 1917–1928 (2005)
16. Lazos, L., Poovendran, R., Čapkun, S.: ROPE: Robust position estimation in wireless sensor networks. In: Proceedings of the 4nd International Symposium on Information Processing in Sensor Networks (IPSN'05), pp. 324–331 (2005)
17. Liu, D., Ning, P., Du, W.: Detecting malicious beacons for secure localization discovery in wireless sensor networks. In: Proceedings

of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), pp. 609–619 (2005)
18. Zhou, Y., Fang, Y., Zhang, Y.: Securing wireless sensor networks: a survey. IEEE Commun. Surv. Tutor. **10**, 6–28 (2008)
19. Liu, D., Ning, P.: Location-based pairwise key establishments for static sensor networks. In: Proceedings of the 1st ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN'03), pp. 72–82 (2003)
20. Du, W., Deng, J., Han, Y.S., Chen, S., Varshney, P.K.: A key management scheme for wireless sensor networks using deployment knowledge. In: Proceedings of the 23rd Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04), pp. 586–597 (2004)
21. Liu, D., Ning, P.: Improving key predistribution with deployment knowledge in static sensor networks. ACM Trans. Sens. Netw. **1**, 204–239 (2005)
22. Younis, M.F., Ghumman, K., Eltoweissy, M.: Location-aware combinatorial key management scheme for clustered sensor networks. IEEE Trans. Parallel Distrib. Syst. **17**, 865–882 (2006)
23. Misra, S., Bhardwaj, S., Xue, G.: ROSETTA: Robust and secure mobile target tracking in a wireless ad hoc environment. In: Proceedings of the IEEE Military Communication Conference 2006 (MILCOM'06), pp. 1–7 (2006)
24. Li, Z., Trappe, W., Zhang, Y., Nath, B.: Robust statistical methods for securing wireless localization in sensor networks. In: Proceedings of the 4nd International Symposium on Information Processing in Sensor Networks (IPSN'05), pp. 91–98 (2005)
25. Zhong, S., Jadliwala, M., Upadhyaya, S., Qiao, C.: Towards a theory of robust localization against malicious beacon nodes. In: Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM'08), pp. 1391–1399 (2008)
26. He, D., Cui, L., Huang, H., Ma, M.: Design and verification of enhanced secure localization scheme in wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. **20**, 1050–1058 (2009)
27. Zhang, Y., Liu, W., Fang, Y., Wu, D.: Secure localization and authentication in ultra-wideband sensor networks. IEEE J. Sel. Areas Commun. **24**, 829–835 (2006)
28. Zhu, W.T., Xiang, Y.: Argus: A light-weighted secure localization scheme for sensor networks. In: Proceedings of the 6th International Conference on Autonomic and Trusted Computing (ATC'09), Lecture Notes in Computer Science, vol. 5586, pp. 164–178 (2009)
29. Chan, H., Perrig, A., Song, D.: Random key predistribution schemes for sensor networks. In: Proceedings of the 24th IEEE Symposium on Security and Privacy (SSP'03), pp. 197–213 (2003)
30. Du, W., Deng, J., Han, Y.S., Varshney, P.K.: A pairwise key predistribution scheme for wireless sensor networks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03), pp. 42–51 (2003)
31. Yang, W., Zhu, W.T.: Voting-on-grid clustering for secure localization in wireless sensor networks. In: Proceedings of the IEEE International Conference on Communications (ICC) 2010, (2010)