Locating Malicious Nodes for Data Aggregation in Wireless Networks

Xiaohua Xu*, Qian Wang†, Jiannong Cao‡, Peng-Jun Wan*, Kui Ren†, Yuanfang Chen§
*Department of Computer Science, Illinois Institute of Technology, Chicago, IL
†Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL
‡Department of Computing, The Hong Kong Polytechnic University, Hong Kong
§School of Software, Dalian University of Technology, P.R.China

***Abstract*—Data aggregation, as a primitive communication task in wireless networks, can reduce the communication complexity. However, in-network aggregation usually brings an unavoidable security defect. Some malicious nodes may control a large percentage of the whole network data and compel the network misbehave in an arbitrary manner. Thus, locating the malicious nodes to prevent them from further disaster is a practical challenge for data aggregation schemes. Based on the grouping and localization techniques, we propose a novel integrated protocol to locate malicious nodes. The proposed protocol does not rely on any special hardware and requests only incomplete information of the network from the security schemes. We also conduct simulation study to evaluate the proposed protocol.**

I. Introduction

Wireless networks, composed of a large number of spatially distributed static or mobile wireless devices, are increasingly being deployed and used for a variety number of purposes, from environmental monitoring, to critical infrastructure protection, to health-care, to precise agriculture. The intrinsic characteristic limitations of wireless systems such as power restrictions, scarce computational power and storage raise practical issues for wireless networking applications. To make a wireless networking application successful, a number of (theoretical and/or practical) challenging issues must be addressed, such as deployment strategies, energy conservation, efficient data processing, routing, and localization. Among them, one practical and fundamental challenge is efficient data processing. As we know, data are generated everywhere in wireless networks. In most applications, there is a request to send all data from the wireless nodes within the network to a control center (or sink node) that has more computational ability than other wireless nodes. This process is termed as converge-cast. Different from data collection, data aggregation allows in-network processing which means that data can be compressed within the network. This feature introduces a possibility of a new energy or time efficient method of collecting data, comparing to raw data collection.

From the viewpoint of information theory, data aggregation is a lossy data compression process because all the individual sensory readings are lost in the per-hop aggregation process. However, from the security perspective, data aggregation opens a new door to false data injection attacks. As we know, wireless nodes are often deployed in open and unattended environments. Due to the low manufacturing cost, the nodes cannot prevent physical tampering. During data aggregation, an adversary can obtain the confidential information from a compromised node (or jammer). The compromised node may also report an arbitrary false fusion result, causing the final aggregation result to far deviate from the true measurement. This attack can become more damaging when multiple compromised nodes collude in injecting false data. Even worse case is that the adversary reprogram the jammer(s) with malicious code. The malicious node can change the aggregation result of a large part of the whole network and cause inconceivable consequences. To prevent possible disasters, the sink node should know the locations of jammers immediately to obtain a correct result on the premise of efficiency for data aggregation.

However, efficiency and security are two complementary objectives (tradeoff) for data aggregation. An efficient data aggregation scheme cannot have a good performance on security issues, which means that it cannot provide very detailed information about the attack, such as the concrete geographical information. By the aid of some localization scheme (led by the high-cited work [3]), we can obtain the concrete geographical information, then it is easy for us to deal with the attack (isolate either all the data in the area or the affected data *w.h.p*). Based on this insight, we will propose an efficient but secure data aggregation scheme. We first divide the whole network into groups, perform aggregation in each logical group, and generate one aggregate from each group. After the sink node collects all the group aggregates, it identifies the malicious groups based on an outlier detection algorithm. Here a malicious group is defined as a group containing malicious nodes. We then locate malicious nodes based on the incomplete information acquired by identifying malicious groups. Thus, our protocol integrates both detecting and locating of malicious nodes.

Note that most data aggregation schemes adopt a tree routing structure [10], [13], then a malicious node closer to the sink node could access a large percentage of the whole network data and would have a larger impact on the manipulation of final result as observed in [14]. Our protocol can reduce the trust on high-level nodes and perceive all low-cost wireless nodes as evenly trustable, which is realized by the principle of divide-and-conquer (or grouping). Our grouping technique can dynamically partition the network into multiple groups of similar sizes. Since fewer nodes will be under a high-level

978-1-4673-0775-8/12/$31.00 ©2012 IEEE

node in a logical subtree, the potential security threat by a malicious high-level node will be reduced.

The rest of this paper is organized as follows. Section II formulates the problem to be studied. Section III is devoted to the protocol design. Section IV presents our simulation results. Section V outlines the related work. Finally, Section VI concludes the paper.

II. Network Model

Given a wireless network of n nodes $V = \{v_1, v_2, \cdots, v_n\}$; in addition, there is a distinguished sink node v_s that connects this network to the outside infrastructure such as the Internet. Each node can monitor the environment, and collect some data. Assume that $A = \{a_1, a_2, \cdots, a_N\}$ is a totally *ordered multi-set* of N data items collected by all n nodes v_i, $1 \leq i \leq n$, at a certain time period. Here, N is the cardinality of set A. Each node v_i has n_i amount of raw data, denoted as $A_i \subset A$. Since A is a multi-set, we assume $A_i \cap A_j = \emptyset$ for $i \neq j$ and $A = \bigcup_{i=1}^{n} A_i$. Then $\langle A_1, A_2, \cdots, A_n \rangle$ is called a distribution of A at sites of V. We assume that one packet (*i.e.*, message) can contain one data item a_i, the node ID, plus an additional constant number of bits, *i.e.*, the packet size is at the order of $\Theta(\log n + \log U)$, where U is the upper-bound on values of a_i. We also assume there is a reliable transmission mechanism, for example, by using a link-layer acknowledgment protocol. *Data aggregation* is a process where all data are gathered to the sink node and expressed in a summary form by using some aggregation function. Here aggregation functions can be classified into three categories: distributive (*e.g.*, *maximum, minimum, sum, count*), algebraic (*e.g.*, *minus, average, variance*) and holistic (*e.g.*, *median, k^{th} smallest or largest*). The previous two categories are usually the focus of network community. An aggregation function f is said to be *distributive* if for every pair of disjoint data sets X_1, X_2, we have $f(X_1 \cup X_2) = f_g(f(X_1), f(X_2))$ for some function f_g. An algebraic aggregation function f can be expressed as a combination of k distributive functions for some integer constant k, *i.e.*,

$$f(X) = f_g(g_1(X), g_2(X), ..., g_k(X)).$$

Thus, instead of computing f, we compute $g_i(X)$ distributively for $i \in [1, k]$ and $f_g(g_1, g_2, \cdots, g_k)$ at sink node. The detailed definition of aggregation function is available in [13].

While message authentication code (MAC) can easily defeat an outsider adversary from launching many attacks, there still exist some attacks such as behavior-based attacks and false data injection attacks. We will focus on the latter one where an attacker aims to furtively inject false values that deviate from the true measures in a noticeable scale. In the context of data aggregation, the attack could be forging an unusual false data value or forging a large count value (the number of nodes involved in the aggregation operation). Note that an attacker may launch these two attacks simultaneously. A node is not perceived to attack when it forges a false reading of its own. We want to defend against the false data injection attacks which compel the sink node accept false aggregation results. Given a wireless network of n nodes, with some malicious nodes, the objective is to find out these malicious nodes efficiently to isolate them with small communication overhead.

III. Protocol Design

A. Grouping and leader selection

We geometrically partition the deployment plane into triangles. The wireless nodes which lie in the same triangle form a group. During the partition process, the size of each triangle varies to balance the group size.

We share the same group leader selection criteria with [14] with two noticeable modifications. First, all their groups are logical while we use physical partition of the topology tree in order to apply a localization technique. Second, they need to employ hop-by-hop verification to find out the compromised nodes while our protocol save that effort. Third, their topology tree is a data structure based on a real topology tree. Their tree is fixed all the time which is vulnerable to attacks while our design will rotate the leaders among nodes instead of fixing their roles. By doing this, we can ensure that the attackers cannot predict the group leaders *w.h.p.* Otherwise, the attacker can target at the group leaders and compromise them. Another benefit is that each time, every node is assigned into a different group that is formed on the fly; we can balance the resource usage of nodes so as to prolong the overall lifetime of the network.

B. Aggregation commitment

Leaf node aggregation: A leaf node just sends its identification, data, and count value to its parent (it also keeps a local copy of the packet). The packet consists of the data as follows. First, there is a flag indicating whether the data can be aggregated or not (the value of bit 1 indicates that the data can not be aggregated). This flag position is reserved for later usage. Second, there is a count value indicating how many nodes' data the packet contains. Third, there is a node reading. Finally, there is an authentication value (encrypted) computed by the leaf node with its individual key shared with the sink node. In addition, a seed is included to identify this specific data aggregation process and to prevent replay attacks. All the data will be encrypted using its pairwise key shared with the parent of this leaf node.

Internal node aggregation: When an internal node receives a packet from its child node, it first checks the flag. If the flag bit is 0, the node first decrypts the data using its pairwise key shared with this child node. It also performs some simple checking on the validity of the count (if within a certain range), and seed. If the packet does not pass this checking, it will discard the packet directly. Otherwise, it will further aggregate its own reading with all the aggregates carrying flag 0 received from its child nodes. A new count is also calculated as the sum of the count values in the received aggregates with flag 0 plus one (accounting for its own reading).

The internal node also checks if it is a group leader, it then encrypts the new count value and aggregation data using the pairwise key shared with its own parent. If the internal node is not a leader, then the packet that it sends to its parent node is as follows: the count value summed over the count value of its children and its own contribution, the aggregation value and the XOR of all contributing nodes' authentication values. Finally, all the data will be encrypted using its pairwise key shared with the parent of this internal node. Thus, the encrypted value can represent the authentication information of all the nodes contributing to this aggregation data.

Leader node aggregation: Now suppose that an internal node has processed the aggregates from its child nodes and it finds out that it is a group leader. Like a regular internal node, it also computes a new aggregate, keeps local copies of those packets with flag bit 0, and appends a corresponding authentication encrypted value using its individual key shared with the sink node. Unlike a regular internal node, it sets the flag to 1 in its aggregation packet so that data from this group will not be aggregated any more. The packet it sends upward is as follows: the aggregation result of the group, the authentication value computed by the leader node. Similarly, all the data will be encrypted using its pairwise key shared with the parent of this leader node.

When the parent node receives a packet from a leader node, it forwards the packet towards the sink node without any further aggregation. At the same time, this parent node does not add any count value to its own. In an extreme case when all the children of a node are group leaders, this node will only contribute the count value of one to its parent node. In this case, all node behaves like a leaf node.

Based on the above aggregation rule, the packets are transmitted towards the sink node. There may be some nodes left without group membership. In this case, the sink node is the default group leader for them. After the sink node receives the aggregates from all groups, it decrypts and saves them, including the group leader node's identification, the group count, the group aggregation value, the authentication tag computed by the group leader, and the seed.

C. Detecting malicious groups

The next step is to test whether the count or the aggregation result has been modified by a malicious group leader or member, which can influence the final aggregation result at the sink node. Here a malicious group is defined as the group that contains a malicious node. Note that authentication cannot solve this insider attack because a malicious node has the valid keys.

We expect the attacker to forge an aggregated data that have a nontrivial influence on the final result. As a result, a false aggregate should exhibit certain abnormality. On the other hand, we cannot simply treat all abnormal sensing data as outliers and discard them, since they may indeed reflect the real environment. In many cases, we are more interested in abnormal data than in normal ones. We have to verify the abnormal aggregates before accepting or rejecting them.

Grubbs' test [2] Given a data set $X = \{a_1, a_2, \cdots, a_N\}$, suppose that μ and s are the sample mean and standard deviation of all the data, then the data a_i with the largest sample statistic

$$Z = \frac{|a_i - \mu|}{s} \tag{1}$$

is an outlier if this statistic falls beyond the range defined by the critical values.

In Grubbs' test, it first computes the sample statistic for each datum a_i in the set by $\frac{|a_i-\mu|}{s}$. The result represents the datum's absolute deviation from the sample mean in units of the sample standard deviation. Based on this, each time the datum with the maximum statistic is picked up. Then we check whether the sample statistic falls in the non-rejection range defined by the critical values. Therefore we can use Grubbs' test to determine that which groups are malicious groups and thus contain malicious nodes. We next show how to locate the malicious nodes as accurately as possible.

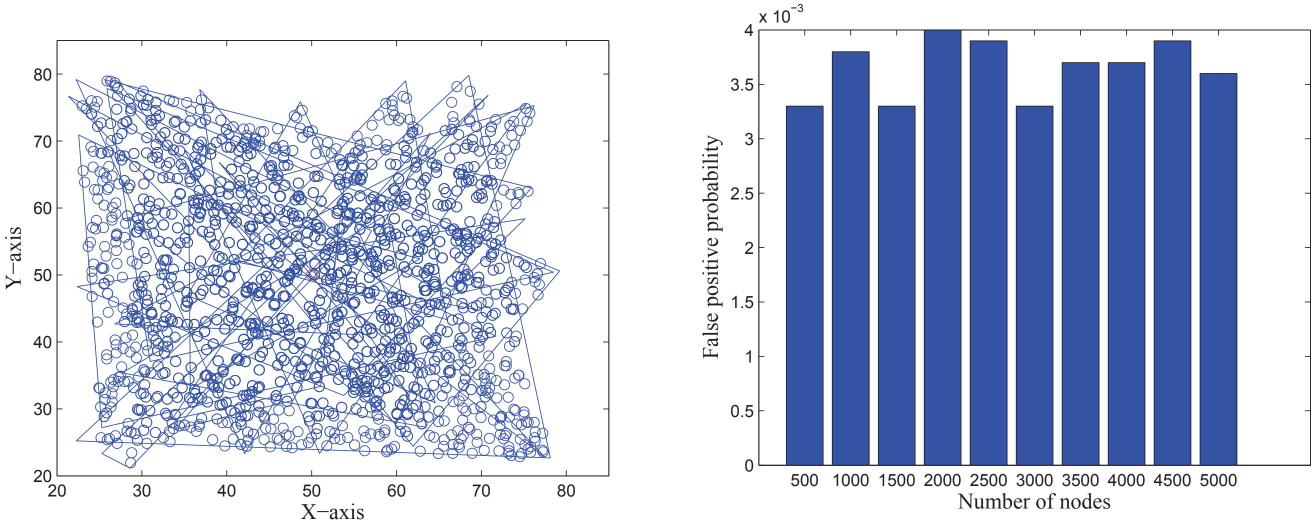
To detect multiple outliers from bivariate data (*i.e.*, counts and aggregation value), a simple method is that, after Grubbs' test detects one outlier at a time, we delete the detected outlier from the data set and repeat the test over the remaining data until no outliers can be found.

D. Locating malicious node

Section III-C illustrate the detection of malicious groups. By performing grouping several times, we can obtain some incomplete information about which triangular groups the compromised node has appeared in. Based on this incomplete information, we want to locate the malicious node.

APIT [3] is an area-based range-free localization scheme. It employs a novel area-based approach to perform location estimation by isolating the environment into triangular regions between anchor nodes. Any node's presence inside or outside of these triangular regions allows a node to narrow down the area in which it can potentially reside. By utilizing different combinations of triangles, the size of the estimated area in which a node resides can be reduced to provide a satisfying location estimate. Based on APIT, we will repeat grouping with different triangle combinations until all combinations are exhausted or the required accuracy is achieved. We then calculate the center of gravity of the intersection of all of the triangles in which the malicious node resides to determine its estimated position.

If the network size (or the total number of nodes) is large, we can aggregate the results of individual group tests by means of a grid array [3]. This grid array is used to represent the maximum area in which a malicious node likely resides. When we determine the malicious node is inside a particular region, the values of the grids over which the corresponding triangle resides are incremented; the grid area for an outside decision is similarly decremented. Once all triangular regions are computed, the resulting information is used to the maximum overlapping area. Note that some of the inside/outside decisions may be incorrect. However, the correct decisions build up on the grid and the small number of errors only serves as a slight disturbance to the final estimation.



(a) Illustration of grouping.

(b) False positive vs. number of nodes.

Fig. 1. The simulation results of evaluating our proposed protocol.

IV. Simulation Results

We conduct simulations to study the performance of our protocol. We deploy a network of 5000 wireless nodes in a two-dimensional plane, each node contains data to report. We set one node to be malicious. Thus the value of this node deviates from that of other nodes greatly. The objective is to find this node efficiently during the secure aggregation process. We will determine the false positive probability.

To implement our protocol, we will group all nodes for 20 times. Each time, every group is a triangle containing all nodes in this triangle. We will vary the shape of each triangle extensively to ensure that the intersection areas of different triangles are small. Then, each group will check whether itself is a non-secure group. After grouping all nodes for 20 times, we can find 20 non-secure groups. We then find the overlapping area of the 20 non-secure groups. The center of gravity for this overlapping area is highly suspicious.

Figure 1 (a) shows the result of a network of 5000 nodes after grouping for 20 times. Each time when we group, the shape of each triangle varies.

Figure 1 (b) shows the result of false positive rate versus the number of nodes. We set the number of grouping times as 20. The result is promising. Most of the time, we can find the malicious node with high probability. At the same time, we can see that the false positive rate is not very sensitive to the number of nodes.

V. Related Work

Recently many data aggregation protocols have been proposed to eliminate the data redundancy in sensory data of the network, hence reducing the communication cost and energy expenditure in data collection. For the well-studied minimum latency data aggregation problem, there is a series of results focusing on either the protocol wireless interference model [5], [10], [13] or the physical interference model [11], [12]. There is also a recent work [6] for data aggregation scheduling in duty-cycled network.

For secure data aggregation, Ozdemir *et al.* [7] present a broad overview of secure data aggregation by evaluating each protocol based on the security requirements of wireless sensor networks. In [4], the authors *et al.* proposed a protocol called iCPDA, which piggybacks on a cluster-based privacy-preserving data aggregation protocol(CPDA). They implement the add-on feature to protect integrity of aggregation result.

We focus on a general practical challenge: how can the sink node know the location of compromised nodes and thus obtain a correct aggregation result without losing the efficiency of per-hop data aggregation when a fraction of sensor nodes are compromised? For this problem, Wagner [9] first addressed it and provided guidelines for selecting aggregation functions in a sensor network. Yang *et al.* [14] proposed SDAP, a secure hop-by-hop data aggregation protocol using a tree-based topology to compute the Average in the presence of a few compromised nodes. SDAP divides the network into multiple groups and employs an outlier detection algorithm to detect the corrupted groups. Recently, Roy *et al.* [8] also used a grouping technique in their extended approach for secure median computation. Chen *et al.* [1] proposed to strictly diminish the capability of adversaries whenever they launch a successful attack, so that malicious sensors can only ruin the aggregation result for a small number of times before they are fully revoked. To this end, they proposed VMAT (verifiable minimum with audit trail), a novel secure aggregation protocol with malicious sensor revocation capability.

VI. Conclusion

We proposed a two-phased integrated protocol for detecting and locating compromised nodes in wireless networks. In the first phase, we dynamically grouped the wireless nodes and performed aggregation commitment in each group before the sink node collects the data. After collecting all the aggregation commitment to the sink node, we then detected the malicious groups based on the Grubb's test scheme. In the second phase, we located compromised nodes based on the incomplete information acquired from the first phase. By the aid of some localization scheme, we obtain the concrete geographical information of the attack. While the proposed secure data aggregation scheme is efficient after performance evaluation, it will be interesting to develop efficient schemes in the general context (independent of Grubb's test).

VII. Acknowledgement

Peng-Jun Wan and Xiaohua Xu were supported in part by the NSF grants CNS-0831939 and CNS-0916666.

Kui Ren and Qian Wang were supported in part by the NSF grants CNS-0831963 and CNS-1117811.

References

[1] Chen, B., and Yu, H. Secure aggregation with malicious node revocation in sensor networks. *IEEE ICDCS* (2011).

[2] Grubbs, F. Procedures for detecting outlying observations in samples. *Technometrics*, (1969).

[3] He, T., Huang, C., Blum, B., Stankovic, J., and Abdelzaher, T. Range-free localization schemes for large scale sensor networks. In *ACM Mobicom* (2003).

[4] He, W., Liu, X., Nguyen, H., and Nahrstedt, K. A Cluster-Based Protocol to Enforce Integrity and Preserve Privacy in Data Aggregation. In *IEEE ICDCS Workshops* (2009).

[5] Huang, S., Wan, P., Vu, C., Li, Y., and Yao, F. Nearly Constant Approximation for Data Aggregation Scheduling in Wireless Sensor Networks. In *IEEE INFOCOM* (2007), pp. 366–372.

[6] Jiao, X., Lou, W., Wang, X., Cao, J., Xu, M., and Zhou, X. Data aggregation scheduling in uncoordinated duty-cycled wireless sensor networks under protocol interference model. To appear at *Ad-hoc & Sensor Wireless Networks*.

[7] Ozdemir, S., and Xiao, Y. Secure data aggregation in wireless sensor networks: A comprehensive overview. *Computer Networks* (2009).

[8] Roy, S., Conti, M., Setia, S., and Jajodia, S. Secure median computation in wireless sensor networks. *Ad Hoc Networks* (2009).

[9] Wagner, D. Resilient aggregation in sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks* (2004).

[10] Wan, P., Scott, C., Wang, L., Wan, Z., and Jia, X. Minimum-latency aggregation scheduling in multihop wireless networks. In *ACM Mobihoc* (2009).

[11] Wan, P., Wang, L., and Frieder, O. Fast group communications in multihop wireless networks subject to physical interference. In *IEEE MASS* (2009).

[12] Xiang-Yang Li, XiaoHua Xu, S. W. S. T. G. D. J. Z. Y. Q. Efficient Data Aggregation in Multi-hop Wireless Sensor Networks under Physical Interference Model. In *IEEE MASS* (2009).

[13] Xiaohua Xu, Shiguang Wang, X. M. S. T. X. L. An Improved Approximation Algorithm for Data Aggregation in Multi-hop Wireless Sensor Networks. In *IEEE TPDS* (2011).

[14] Yang, Y., Wang, X., Zhu, S., and Cao, G. SDAP: A secure hop-by-hop data aggregation protocol for sensor networks. *ACM Transactions on Information and System Security* (2008).