Information Sciences 254 (2014) 39–53

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Defending collaborative false data injection attacks in wireless sensor networks

Jianxin Wang [a,*], Zhixiong Liu [a,b], Shigeng Zhang [a], Xi Zhang [c]

[a] School of Information Science and Engineering, Central South University, Changsha, Hunan, China
[b] Department of Computer Science and Technology, Changsha University, Changsha, Hunan, China
[c] Department of Electrical and Computer Engineering, Texas A&M University, TX, USA

ARTICLE INFO

Article history:
Received 7 March 2012
Received in revised form 11 July 2013
Accepted 5 August 2013
Available online 20 August 2013

Keywords:
Wireless sensor network
Compromised node
False data injection
Location information
Relative position

ABSTRACT

False data filtering is an important issue in wireless sensor networks. In this paper, we consider a new type of false data injection attacks called collaborative false data injection, and propose two schemes to defend such attacks. In collaborative false data injection attacks, multiple compromised nodes collaboratively forge a fake report and inject the report into the network. This type of attacks is hard to defend with existing approaches, because they only verify a fixed number of message authentication codes (MACs) carried in the data report but the adversary can easily obtain enough compromised nodes from different geographical areas of the network to break their security. Our novel solution is to bind the keys of sensor nodes to their geographical locations, and verify the legitimacy of a data report by checking whether the locations of the sensors endorsing the report are logical (e.g., the sensors should be close enough to each other to sense the same event). We propose two filtering schemes: The geographical information based false data filtering scheme (GFFS) which utilizes the absolute positions of sensors in the verification, and the neighbor information based false data filtering scheme (NFFS) which utilizes relative positions of sensors when absolute positions cannot be obtained. We theoretically analyze the filtering probability of the two proposed schemes, and evaluate their performance through extensive simulations. Simulation results show that, when there are totally ten nodes compromised in a 400 nodes network, the detection probability of collaborative false data injection attacks is higher than 97% in GFFS and NFFS, but is less than 7% in traditional false data filtering approaches such as SEF.

1. Introduction

Wireless sensor networks (WSNs) are widely used in many applications including military surveillance, habitat monitoring, and health care [14]. WSNs are usually composed of a large amount of sensor nodes with limited resources, and are usually deployed in unattended environments. In such environments, the security of sensor nodes is very important [7,8,10,13,17,24,25]. Once a node is compromised, the adversary will disclose all the secret information stored in that node. The adversary can then use the compromised nodes to launch false data injection attacks [16], i.e. to inject bogus reports into sensor networks. Defending false data injection attacks is an important research issue in WSNs, because this type of attacks not only causes false alarms that waste real-world response efforts (e.g. sending response teams to the event location), but also may drain out the constrained resources of the forwarding sensors.

* Corresponding author. Tel./fax: +86 0731 88830212.
E-mail addresses: jxwang@mail.csu.edu.cn (J. Wang), lzxterry@163.com (Z. Liu), sgzhang@mail.csu.edu.cn (S. Zhang), xizhang@ece.tamu.edu (X. Zhang).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.08.019

To prevent such false data injection attacks, a general en-route filtering framework has been proposed to detect and filter out false reports in [23]. In this framework, every node is preloaded with some symmetric keys. When an event happens, multiple surrounding nodes collaboratively generate a report that carries $t(t > 1)$ distinct message authentication codes (MACs). Here t is a security threshold. A MAC represents a node's agreement on the report, which is generated by using one of the symmetric keys stored in that node. During the forwarding of the report towards the sink, each forwarding node verifies the correctness of the MACs carried in the report in a probabilistic manner. A report that carries less than t MACs or contains wrong MACs is detected as an invalid report, and hence will be dropped by intermediate nodes or the sink. The framework proposed in [23] inspired some following researches on false reports filtering in recent years [1,15,18–21,27–30]. Most of them focus on improving the filtering probability and reducing the energy consumption of sensor nodes.

However, existing data filtering schemes only consider whether there is *enough number* of sensors endorsing the data report or not. They do not consider whether the endorsement of these sensors is *logical* or not. This makes them fail in filtering out false data report forged collaboratively by more than t compromised nodes from different geographical areas. An example of collaborative false data injection attack is illustrated in Fig. 1. In this example, the adversary has compromised five nodes $S_1, \ldots, S_5$. Assume that the security threshold is five and the compromised nodes have distinct key partitions. By coordinating the five compromised nodes, the adversary can successfully claim a fabricated event at an arbitrary location and forge a data report. Existing data filtering schemes, e.g. SEF, will fail to correctly filter out this data report because there is enough number of correct MACs in the data report. On the other hand, if we take the locations of the endorsing sensors of the data report into account, we can correctly find that the data report is fake because it is not logical: The five sensors endorsing the event are far from each other, so they cannot observe the event simultaneously.

In this paper, we study collaborative false data injection attack and propose two schemes to defend this type of attacks. The novelty of our schemes is that we bind the keys of sensor nodes to their geographical locations. In the geographical information based false data filtering scheme (GFFS), we assume that sensor nodes know their absolute geographical locations and utilize this information to filter out fake reports forged by compromised sensors from different geographical areas. Considering that GFFS requires expensive positioning devices (e.g., GPS), we then propose a neighbor-information based false data filtering scheme (NFFS). NFFS utilizes relative positions of sensor nodes to defend collaborative false data injection attacks. Theoretical analysis and simulation results show that GFFS and NFFS can effectively resist the collaborative false data injection attacks, and tolerate much more compromised nodes than existing schemes.

The main contributions of this paper are summarized as follows:

First, we propose a new false data injection model called collaborative false data injection, and point out that existing data filtering approaches cannot defend such attacks. To our knowledge, we are the first to investigate collaborative false data injection attacks in wireless sensor networks.

Second, we propose the GFFS scheme. In GFFS, each node distributes its location information to some forwarding nodes after deployment. Each data report must carry the MACs and locations of t detecting nodes that sense the event simultaneously. All the forwarding nodes verify the correctness of both MACs and locations. Besides, they also verify the legitimacy of the t locations. Because the keys of sensor nodes are bound to their geographical locations, false reports injected collaboratively by compromised nodes from different geographical areas can be detected and filtered out. Moreover, the ability of compromise tolerance can also be enhanced. Simulation results show that when ten nodes are compromised in a network containing 400 nodes, the probability that the adversary breaks down GFFS is only 3%, while it can break down the security of SEF with a probability of 93.2%.

Third, considering that GFFS requires expensive positioning devices, we further propose the NFFS scheme. In NFFS, each node distributes its neighbor information to some other nodes after deployment. When a report is generated for an observed event, it must carry the IDs and MACs from t detecting nodes. Each forwarding node checks the correctness of the MACs carried in the report and the legitimacy of relative positions of detecting nodes. As a result, false data reports can be detected by

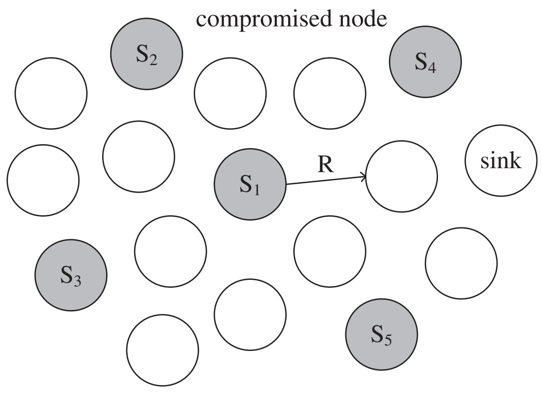

Fig. 1. An example of collaborative false data injection attack.

checking the relationship between the keys of sensor nodes and their locations. Consequently, the security protection against compromised nodes can be enhanced greatly. Simulation results show that when ten nodes are compromised in a network containing 400 nodes, the probability that the adversary breaks down NFFS is only 0.7%, even smaller than that in GFFS. However, the storage cost in NFFS is higher than in GFFS.

The rest of the paper is organized as follows. We survey related work on false data filtering in Section 2. We then describe our system model and threat model in Section 3. We describe the detailed design of GFFS and NFFS in Sections 4 and 5, respectively. We also theoretically analyze the performance of GFFS and NFFS in these two sections. In Section 6 we analyze the security of the two proposed schemes. In Section 7 we report our simulation results. Finally, we conclude this paper in Section 8.

2. Related work

According to the encrypting methods, existing researches on false data filtering fall into two categories: symmetric key based schemes [19,21,23,29] and asymmetric key based schemes [1,4,12,15,18,20,26]. Asymmetric cryptographic algorithms are computation intensive [29], thus are not suitable to be used on sensor nodes that have extremely limited computational capacity and cannot afford high computational overhead. Most of existing data filtering schemes for WSNs are based on symmetric key. Some researchers even argue that the symmetric key technique is possibly the only practical approach to establishing secure channels among sensor nodes.

One pioneering work in en-route false report filtering is SEF proposed by Fan et al. [23]. In SEF, a key pool is divided into $n(n > t)$ partitions, each of which contains m keys. Each node randomly picks k keys from one partition. When an event occurs, t nodes sensing the event simultaneously collaboratively generate a legitimate report. In the filtering phase, each forwarding node verifies the correctness of a MAC in the report with a probability of k/m. The sink serves as the final guard to filter out false reports. SEF is independent with data dissemination protocols. However, it cannot defend collaborative false data injection attacks. An adversary can easily break down SEF's security protection by compromising t nodes from different geographical areas and forging a report collaboratively. Furthermore, in SEF false reports may travel many hops before they are filtered out. This wastes the constrained energy of the forwarding nodes.

Zhu et al. [29] proposed an interleaved hop-by-hop filtering scheme IHA, in which each node is associated with two other nodes on the path from the cluster head to the sink. The two nodes are called the lower associated node and upper associated node, respectively. An en-route node will verify the correctness of the MAC generated by its lower associated node. Upon successful verification, it replaces the old MAC with a new one using its pair-wise key shared with its lower associated node. IHA is able to filter the false reports within t hops. However, association between nodes may break down due to frequent routing changes, and the filtering probability decreases greatly when this happens.

Yu et al. [19] presented a dynamic en-route false report filtering scheme called DEFS. In DEFS, all the nodes are organized into clusters after deployment. Each cluster head then assigns keys to its upstream nodes on several different paths by using the *Hill Climbing* algorithm. In the filtering phase, each forwarding node verifies the authenticity of the reports and drops false ones. The hill climbing approach guarantees that, for a cluster, sensors closer to the cluster hold more verifying keys than sensors far from it. As a result, DEFS is able to lighten the overhead of nodes close to sink during reports forwarding. However, DEFS consumes a large amount of energy in key distribution. Moreover, multi-path based forwarding is not suitable for energy limited sensor networks.

Yu et al. [21] proposed a grouping-based filtering scheme named GRSEF. In GRSEF, nodes are divided into exact t groups rather than n $(n > t)$ groups after deployment. This ensures that any position in the network can be covered by t nodes from distinct groups with a high probability. The nodes then fetch keys along a multiple axes-based method. GRSEF is independent of sink stationary and routing protocols. It can provide a suitable en-route filtering solution to sensor networks with mobile or stationary sinks. However, multiple axes-based key derivations and maintenance will incur great communication cost.

Yang et al. [20] proposed CCEF, which is based on commutative cipher. In CCEF, the source node establishes a secret association with the base station, and intermediate forwarding nodes use the witness key to verify the authenticity of the reports without knowing the original session key. CCEF achieves stronger security protection than existing symmetric key sharing approaches. Wang et al. [18] used the elliptic curve cryptography to further improve the security of CCEF. Ayday et al. [1] proposed a location-aware network coding security mechanism called LNCS, which employs random network coding to provide data availability. Based on LNCS, Ren et al. [15] proposed a location-aware end-to-end security framework called LEDS, which works only in some particular routing schemes.

Yang et al. [26] proposed a multi-dimensional resilient statistical en-route filtering scheme called MDSEF. In MDSEF, the overall key pool is divided into multiple sets. Each set is composed of a number of groups, and each group is composed of a number of keys. Each node could join one group in each set and obtain some keys from these groups. Since each node obtains keys from multiple groups, the covering performance and filtering effectiveness of the mechanism can be improved simultaneously. An axis-rotation method is proposed to associate the terrain with multiple sets and implement location-based key derivation. Moreover, a distributed group joining method with stepwise refinement is proposed for nodes' group selection. Analysis and simulation results show that MDSEF could significantly improve the covering performance and filtering effectiveness without losing the resiliency against node compromise.

Naresh et al. [12] presented an active en-route filtering scheme for information reporting called AEFS, which aimed to deal with DoS attacks and false data injection attacks simultaneously in wireless sensor networks. In AEFS, each node has a hash chain of authentication keys that are used to endorse reports. Meanwhile, a legitimate report should be authenticated by a certain number of nodes. First, each node disseminates its key to forwarding nodes. Then, after sending reports, the sending nodes disclose their keys, allowing the forwarding nodes to verify their reports. AEFS utilizes the Hill Climbing key dissemination approach to ensure that the nodes closer to data sources have stronger filtering capacity. Moreover, the scheme exploits the broadcast property of wireless communications to defeat DoS attack, and adopts multipath routing to deal with the topology changes. AEFS can drop false reports earlier with a low memory requirement, especially in highly dynamic sensor networks. However, multipath-based information verification will incur great communication cost.

Bashir et al. [4] investigated the duplicate data elimination problem in hybrid RFID-sensor networks. They observed that RFID data contain a lot of duplications. Because RFID data are transmitted to the base station in multi-hop paths, unnecessary transmissions of duplicate data will consume nodes' energy and hence shorten the lifetime of the network. They proposed the RDFS scheme to eliminate duplicated data during the transmission. RDFS used a clustering mechanism, in which cluster heads eliminate duplicate data and forward filtered data towards the base station. Compared with existing filtering schemes, RDFS saves energy in both communication and computation.

Yang et al. [22] presented a polynomial-based compromised-resilient en-route filtering scheme called PCREF. PCREF can filter false injected data effectively and achieve high resilience to the number of compromised nodes without relying on static routes and node localization. Instead of MACs for endorsing measurement reports, PCREF adopts polynomials to achieve the resilience to attacks. Each node stores two types of polynomials: authentication polynomial and check polynomial derived from the primitive polynomial, which are used for endorsing and verifying the measurement reports respectively. Both theoretical analysis and simulation results show that PCREF achieves better filtering capacity and resilience to a large number of compromised nodes than existing schemes.

Lu et al. [9] proposed a novel bandwidth-efficient cooperative authentication scheme named BECAN to filter injected false data. Based on the characteristics of random graph and the cooperative bit-compressed authentication technique, BECAN scheme can save energy by early detecting and filtering the majority of injected false data with minor extra overheads at the en-route nodes. In addition, the sink needs to check only a very small fraction of injected false data, which largely reduces its burden. Both theoretical analysis and simulation results demonstrate the effectiveness of the proposed scheme in terms of high filtering probability and energy saving.

We can see that aforementioned schemes (SEF, IHA, DEFS, GRSEF, MDSEF, AEFS, RDFS, PCREF and BECAN) only consider whether or not there are *enough number* of sensors endorsing the data report, while they do not consider whether the endorsement of these sensors is *logical* or not. This makes them fail to defend collaborative false data injection attacks. This paper aims to find a solution to resisting the collaborative false data injection attacks in wireless sensor networks.

3. System model and threat model

3.1. System model

We consider a sensor network composed of a large number of sensors with sensing radius r_s and communication radius r_c, respectively. Due to cost constraints, we assume that all the sensors are not equipped with tamper-resistant hardware.

We assume that the sensor nodes are densely deployed, so that each stimulus can be detected by more than t nodes. The detecting sensors collaboratively process the signal and elect one of the nodes as the Center-of-Stimulus (CoS) [23]. Same as in SEF, we assume that all the detecting nodes can directly communicate with the CoS. The CoS collects the detection results from all the detecting nodes and summarizes them to produce a synthesized report on behalf of the group. The report is then forwarded toward the sink, typically traversing multiple hops.

The sink has complete knowledge of all the secret information of all the nodes, e.g. the keys, locations and relative positions. The sink has strong computation and storage capabilities. False reports that sneak through en-route filtering will finally be detected and dropped by the sink.

3.2. Threat model

We assume that the sensor network has a short safe bootstrapping phase after deployment, during which sensor nodes are safe (attackers cannot compromise them) to distribute keys, locations and neighbor information. After the bootstrapping phase, the adversary can compromise multiple sensor nodes, obtaining their security information and taking full control of them. We further assume that the attacker cannot compromise the sink. In this paper, we focus on report fabrication attacks [14,26], in which an adversary injects forged sensing data reports through compromised nodes.

4. The GFFS scheme

GFFS includes four phases: the pre-deployment and bootstrapping phase, the report generating phase, the en-route filtering phase, and the sink verifying phase. We mainly focus on the first three phases since the last one is almost the same as in SEF [23].

4.1. Description of GFFS

4.1.1. Pre-deployment and secure bootstrapping

We use a global key pool $G = \{K_i{:}0 \leqslant i \leqslant N-1\}$, whose size is N and is divided into n non-overlapping partitions $\{U_i, 0 \leqslant i \leqslant n-1\}$. Each partition has m keys ($N = n \times m$). Before deployment, each node first randomly selects one of the n partitions, and then randomly chooses k ($k < m$) keys from the selected partition to store.

After deployment, each node S_i obtains its location L_i: (X_i, Y_i) through GPS or other localization algorithms [9], where (X_i, Y_i) means the coordinate of S_i. Each node S_i then distributes c packets (S_i, L_i, U_i) to the intermediate nodes using a novel multicast algorithm *Bubble-geocast* [19], where U_i denotes the key partition index stored by S_i. *Bubble-geocast* can disseminate a fairly large number of seeds quickly and uniformly over a region. As a result, each node in the network stores the location information for other nodes with a probability c/N_a, where N_a denotes the number of nodes deployed in the whole monitoring area.

As the initialization phase is usually very short, we can assume that no node will be compromised during this phase and packets could be directly sent to the intermediate node in plaintext forms. However, it is also possible that the attacker forge IDs and location information in this phase. To enhance the security in the bootstrapping phase, we can use pair-wise key establishment protocols such as PKPDS [6] to establish a session key between the source node and the selected intermediate node. The PKPDS scheme combines Blom's pre-distribution scheme [2] with a random key pre-distribution method. Before deployment, each sensor node receives a random subset of keys from a key pool. To establish a session key for communication, two nodes find a common key within their subsets and use that key as the shared secret key. The source node can then use this session key to securely transmit the packet to the selected intermediate node.

The PKPDS scheme can well protect the authenticity of location information. It may incur some extra communication overhead in the bootstrapping phase. However, it needs to be executed only once after the nodes are deployed, thus the energy consumption is negligible compared with the energy consumed in package forwarding [11].

4.1.2. Report generation

When an event happens, the CoS gets the location of the stimulus (denoted as L_e) through the following way: First, it calculates the overlapping area (denoted as D_o) of all the detecting nodes' sensing areas, then it randomly picks one point S_i from D_o as the approximate value of L_e.

The CoS then broadcasts its reading e to all the detecting nodes. Upon receiving e, a detecting node S checks whether e is consistent with its own sensing value. If they match within a certain error range, S randomly selects one key to generate a MAC M_i: $K_i(e)$. It then sends its ID, location and MAC to the CoS. The CoS generates a report by attaching some extra information to the event e. The extra information includes the IDs, key indices, MACs and locations of t detecting nodes, and L_e. Note that the t nodes must have distinct key partitions. The final report sent out by the CoS looks like $\{e, L_e;\ i_1, i_2, \ldots, i_t;\ M_{i1}, M_{i2}, \ldots, M_{it};\ j_1, j_2, \ldots, j_t;\ L_{j1}, L_{j2}, \ldots, L_{jt}\}$, where i_t denotes the key index and j_t denotes the node ID.

4.1.3. En-route filtering

Definition 1. Assume that an event e happens and a node S_i detects e with sensing radius r_s. Let the location of S_i and of e be L_i, L_e, respectively. If the distance between L_i and L_e is less than or equal to r_s, we say L_i is *legitimate*. Otherwise, we say L_i is *illegitimate*.

$$|L_i, L_e| \leqslant r_s. \tag{1}$$

As a result of randomized key assignment and the location information pre-distribution, each forwarding node can verify the correctness of a MAC or a location in the report with a certain probability. Moreover, each forwarding node can verify the legitimacy of the locations.

Algorithm 1. En-route Filtering in GFFS

```
/* On receiving report R*/
1. Check that $t$ $\{i_v, M_{iv}\}$ tuples exist in $R$; drop $R$ otherwise.
2. Check the $t$ key indices $\{i_v, 1 \leqslant v \leqslant t\}$ belong to $t$ distinct partitions; drop $R$ otherwise.
3. Check that $t$ $\{j_v, L_{jv}\}$ tuples exist in $R$; drop $R$ otherwise.
4. Check that $(|L_e, L_{jv}| \leqslant r_s, 1 \leqslant v \leqslant t)$; drop $R$ otherwise.
5. If it has one location $L \in \{L_{jv}, 1 \leqslant v \leqslant t\}$ and key partition $U$, it first check the key index $i_v$ in $R$ belongs to $U$; drop $R$
   otherwise. It then check the location in $R$ is the same as $L$; drop $R$ otherwise.
6. If it has one key $K \in \{K_{iv}, 1 \leqslant v \leqslant t\}$, it computes $M = K(e)$ and see if the corresponding $M_{iv}$ is the same as $M$; drop $R$
   otherwise.
7. Send $R$ to the next hop.
```

When a node receives a report R, it first examines whether there are t key indices from distinct partitions, t MACs, t IDs and t locations in R. If not all the three requirements are met, R is dropped. Then it checks the legitimacy of the locations of all the detecting nodes according to Eq. (1). If any of these locations is illegitimate, the report is dropped. Next, if the node possesses any of the t locations and the key partition indices, it checks whether the corresponding key index belongs to the right partition and the correctness of the corresponding location. If the key index does not belong to the right partition or the location is not correct, the report is dropped. Furthermore, if the node possesses any of the keys indicated by the key indices, it re-produces the MAC by using its own key and compares the result with the corresponding MAC carried in the report. The report is dropped if the attached one differs from the locally computed one. If they match exactly, or if this node does not possess any of the t keys or the t locations, the node passes the report to the next hop. The pseudo-code for en-route filtering operations is given in Algorithm 1.

4.2. Performance analysis of GFFS

4.2.1. Collaborative false data injection attacks

In this section we analyze the ability of GFFS to resist collaborative false data injection attacks launched by compromised nodes from different geographical areas. We assume that there are N_a nodes uniformly randomly in a monitoring region D. All the nodes have the same sensing radius r_s.

As mentioned in Section 2, existing schemes only consider whether or not there are *enough number* of sensors endorsing the data report, while they do not consider whether the endorsement of these sensors is *logical* or not. This makes them fail to defend collaborative false data injection attacks. GFFS requires that each forwarding node validates not only the correctness of the MACs, but also the legitimacy of the locations carried in the reports. As a result, the damage of the compromised nodes is localized, and thus false reports injected collaboratively by the compromised nodes from arbitrary areas will be detected and filtered out. For example, assume that the adversary has captured five distinct key partitions stored in $S_1, \ldots, S_5$ and the security threshold t is set at 5 (Fig. 2). If the adversary abuses these nodes to fake a report R: $\{e, L_e;\ i_1, i_2, \ldots, i_5;\ M_{i1}, M_{i2}, \ldots, M_{i5};\ j_1, j_2, \ldots, j_5;\ L_{j1}, L_{j2}, \ldots, L_{j5}\}$, R will be dropped by the first forwarding node. This is because the adversary cannot forge legitimate L_e such that the distance between L_1 and L_e and that between L_4 and L_e are both no larger than r_s, due to the fact that the distance between S_1 and S_4 is larger than $2r_s$. So whatever L_e the adversary forges, it will be treated as illegitimate and the forged report will be dropped.

4.2.2. Compromise tolerance

In order to filter false reports that GFFS cannot detect, the attacker has to capture t distinct key partitions within a $\pi \times r_s^2$ circular area. We assume that nodes $S_6, \ldots, S_{10}$ with distinct key partitions are all within a $\pi \times r_s^2$ circular area D_0, as illustrated in Fig. 2. When $t = 5$ and the adversary has captured $S_6, \ldots, S_{10}$, he can forge t correct MACs and t legitimate locations in a report that can pass the detection of intermediate nodes in GFFS.

Theorem 1. *Assume that the adversary randomly captured N_c ($N_c \geqslant t$) sensor nodes in the network in existing en-route false data filtering schemes such as SEF. Then the probability that the adversary obtains at least t distinct partitions (i.e., at least t out of the N_c captured nodes have distinct key partitions) is*

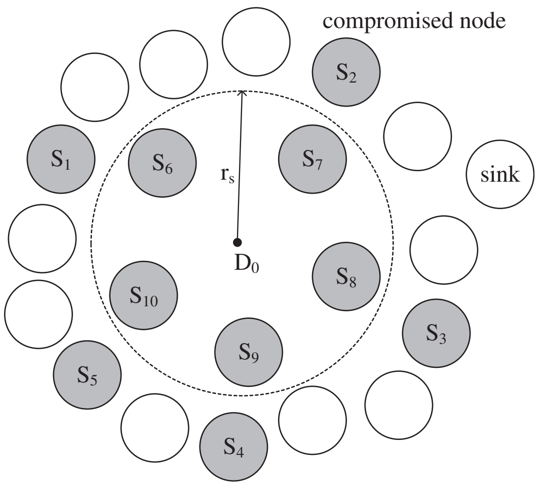

Fig. 2. Collaborative false data injection attack.

$$p_S = \frac{\sum_{i=t}^{n}\left[C_n^i \sum_{j=0}^{i}((-1)^j C_i^{i-j}(i-j)^{N_c})\right]}{n^{N_c}}. \tag{2}$$

Proof. Consider the situation that the adversary gets exact t distinct partitions. First, there are C_n^t ways to take t out of n partitions. Denote the set of these N_c nodes by B_1, and denote the set of the selected t partitions by B_2. Then the number of ways that each node in B_1 takes one partition from B_2 with no partition left unselected in B_2 is

$$Q_S = C_t^t \cdot t^{N_c} - C_t^{t-1}\cdot(t-1)^{N_c} + C_t^{t-2}\cdot(t-2)^{N_c} + (-1)^{t-2}\cdot C_t^2 \cdot 2^{N_c} + (-1)^{t-1}\cdot C_t^1 \cdot 1^{N_c} = \sum_{j=0}^{t-1}((-1)^j C_t^{t-j}(t-j)^{N_c})$$
$$= \sum_{j=0}^{t}((-1)^j C_t^{t-j}(t-j)^{N_c}). \tag{3}$$

Thus the number of ways to get exact t partitions is $C_n^t \times Q_S$. Similarly, we can get the number of ways to get exact $t+1$, $t+2, \ldots,$ and n partitions. Therefore, the total number of ways to get at least t partitions is

$$\sum_{i=t}^{n}\left[C_n^i \sum_{j=0}^{i}((-1)^j C_i^{i-j}(i-j)^{N_c})\right]. \tag{4}$$

Noting that there are n^{Nc} ways for each of the N_c nodes to take one out of n partitions. Consequently, the probability that the adversary can get at least t distinct partitions is P_S. □

Theorem 2. *In GFFS, when N_c sensor nodes are randomly captured in the network ($N_c \geqslant t$), the probability of the adversary to get at least t distinct key partitions within a $\pi \times r_s^2$ circular area is*

$$p_G = \sum_{i=t}^{N_c} \frac{C_{N_c}^i \left(\frac{\pi r_s^2}{D}\right)^i \left(1-\frac{\pi r_s^2}{D}\right)^{N_c-i} A_n^i}{n^i}. \tag{5}$$

Proof. We assume that the $\pi \times r_s^2$ circular area is D_0 in Fig. 2. Consider the situation that the adversary gets exact t distinct key partitions in D_0.

Each node falls in D_0 with probability $\pi \times r_s^2/D$. Then the probability that there are just t nodes falling in D_0 is $p_u = C_{Nc}^t \times (\pi \times r_s^2/D)^t \times (1-\pi \times r_s^2/D)^{Nc-t}$. The probability that each of these t selected nodes has a distinct key partition is $p_r = A_n^t/n^t$. So the probability of the adversary to get just t distinct key partitions in D_0 is $p_u \times p_r$.

Similarly, we can get the probability of the adversary to get just $t+1$, $t+2, \ldots, n$ key partitions in D_0. Therefore, the probability of the adversary to get at least t distinct key partitions within a πr_s^2 circular area is p_G. □

Fig. 3 illustrates the theoretical and simulation results of p_S and p_G when $t = 5$, $D_0/D = 0.25$, and $n = 15$. The simulation results are averaged over 10,000 random tests. From Fig. 3 we can see that, with a small amount of compromised nodes, the adversary is able to break down the security protection of SEF with a high probability. However, in order to break down the security protection of GFFS with a high probability, the adversary has to compromise a large number of compromised nodes. For example, when ten nodes are compromised, the probability that the attacker can break down the security protection of SEF and GFFS is 0.93 and 0.03, respectively. Therefore, both theoretical analysis and simulation results show that GFFS can tolerate much more compromised nodes than existing false data filtering schemes.

4.2.3. Filtering effectiveness

Assume that the number of compromised nodes within a $\pi \times r_s^2$ circular area is N_g ($N_g < t$). In this situation, to produce a seemingly legitimate report, the adversary has to forge $t - N_g$ MACs and $t - N_g$ locations. Then the probability that a forwarding node $S_i$happens to have one of the corresponding $t - N_g$ keys is

$$p_a = \frac{t-N_g}{n}\cdot\frac{k}{m} = \frac{k(t-N_g)}{N}, \tag{6}$$

and the probability that S_i happens to have one of the corresponding $t - N_g$ locations is

$$p_b = 1-(1-c/N_a)^{t-N_g}. \tag{7}$$

Let $P_1 = P_a + P_b - P_a \times P_b$, then the expected fraction of false reports being detected and dropped within h hops is

$$p_h = 1-(1-p_1)^h. \tag{8}$$

The average number of hops that a forged report traverses is given as

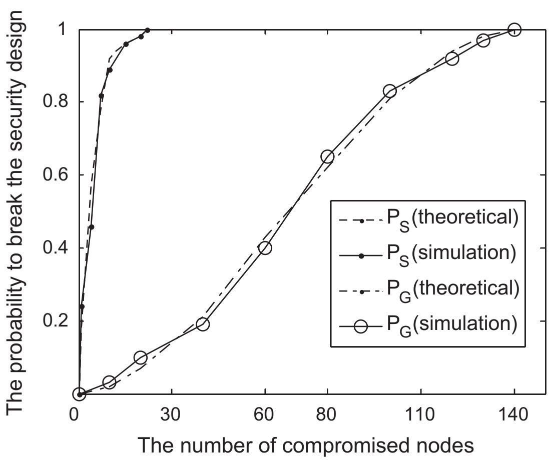

Fig. 3. Theoretical and simulation results of p_S and p_G.

$$H_G = \sum_{i=1}^{\infty} i(1-p_1)^{i-1}p_1 = \frac{1}{p_1}. \tag{9}$$

The number of locations each node stores, *c*, affects the en-route filtering probability and the collaborative injection detecting power. In the worst case, when *c* = 0, the detecting power of GFFS is the same as that in SEF.

4.2.4. Energy consumption

The energy consumption of GFFS comes from four sources. The first is the communication overhead of each node to distribute *c* packets including the location information during the pre-deployment and bootstrapping phase. The second is the communication overhead between the CoS and other detecting nodes when generating a report. The third is the computation overhead of the intermediate nodes during en-route verification. The fourth is the communication overhead of the intermediate nodes during reports forwarding.

As has been pointed out in [11], the energy consumption of MAC computing is much smaller than that of reports transmitting. Moreover, the transmission of small packets between the detecting nodes consumes little energy too. Similarly, the distribution of location information during bootstrapping consumes small amount of energy. Thus we can overlook the three kinds of energy consumption.

Compared with existing false data filtering schemes such as SEF, GFFS requires each report carrying *t* IDs and *t* locations additionally. We use the following model to quantify the energy consumptions. Let the length of a normal report without any extra field, the length of node ID, the length of location and the length of MAC be I_r, I_n, I_k and I_u, respectively. Then, the lengths of a GFFS report and a SEF report become $I_{r0} = I_r + I_k + (I_n + I_k + I_u) \times t$ and $I_{r1} = I_r + (I_n + I_u) \times t$, respectively.

The extra fields in GFFS incur more energy consumption in reports transmitting, reception and computation. However, such extra overhead is affordable because it allows GFFS to resist the coordinated false data injection attacks by the compromised nodes from different geographical areas. On the other hand, GFFS saves energy through its early detection and dropping of false reports while the low detection probability of SEF makes it consume more energy than GFFS. Our simulation results verified this.

4.2.5. Storage overhead

In SEF each node t needs to store *k* keys, while in GFFS each node needs to store extra *c* locations. Assume the size of a key and a location is 64 bits and 10 bits, respectively. To store 50 keys and 60 locations in GFFS, each node incurs about 0.5 KB storage overhead, slightly larger than that in SEF (0.4 KB). Considering that GFFS is able to resist the collaborative false data injections, such extra storage overhead is reasonable. Current mainstream sensor nodes can meet the requirements of GFFS. (E.g., the MICA2 platform is equipped with 4 KB SRAM and 128 KB ROM).

5. NFFS scheme

GFFS requires each sensor be aware of its positions, which need support of expensive positioning devices (e.g., GPS). In this section, we consider utilizing the relative positions of sensor nodes to defense the collaborative false data injection attacks. We design a neighbor information based false data filtering scheme, NFFS. NFFS also includes four phases as in GFFS. We only present their differences in each phase.

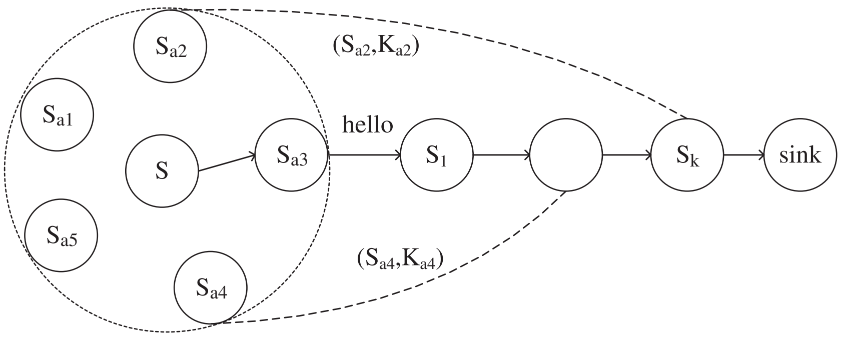

Fig. 4. Distribution of neighbor information and keys.

5.1. Description of NFFS

5.1.1. Pre-deployment and secure bootstrapping

There is a global key pool $G_1 = \{K_i{:}0 \leqslant i \leqslant W\}$. Before deployment, each node S_i selects a distinct key K_i to store.

After deployment, each node S_i first broadcasts a message including its ID and key. Then S_i collects its neighbors' information and generates a *hello* packet $\{S_i,S_{a1},\ldots,S_{aj};\ K_i,K_{a1},\ldots,K_{aj}\}$. Here $S_{a1}, \ldots, S_{aj}$ are neighbors of S_i, and $K_{a1}, \ldots, K_{aj}$ are the corresponding keys of them. After that, S_i establishes a forwarding path to the sink: Path $(S_i) = \{S_i,S_1,\ldots,S_d$, sink}, and transmits *hello* through this path.

On receiving *hello*, each intermediate node $S_k(1 \leqslant k \leqslant d)$ first records S_i's neighbors carried in the packet, and then elects itself as an authentication node of S_i with probability $h_k/\ h_0$, where h_k denotes the number of hops between S_k and the sink, h_0 denotes the number of hops between S_i and sink. If it is successfully elected as an authentication node, S_k stores a randomly selected key $K_{ax}(1 \leqslant x \leqslant j)$ from *hello* and then deletes K_{ax} from *Hello*. Finally, S_k transmits *hello* to the next hop. Otherwise, if it is not elected as an authentication node of S_i, S_k only needs to forward *hello* to the next hop. Fig. 4 illustrates the neighbor information and keys distribution procedure.

Similarly, to enhance the security in the bootstrapping phase, we can establish pair-wise session keys between the source nodes and intermediate nodes. The details are as the same as given in Section 4.A.1).

5.1.2. Report generation

When an event happens, each detecting node S_i uses its key to generate a MAC M_i: $K_i(e)$. The CoS collects the MACs of t detecting nodes and generates a report by attaching the IDs and MACs of these nodes after the event e. The ID of the CoS is put at the first place among all detecting nodes. The final report sent out by the CoS looks like: $\{e;\ S_1,\ldots,S_t;\ M_1,\ldots,M_t\}$, where $S_1$is the CoS.

5.1.3. En-route filtering

Definition 2. Given an event e and the set of detecting nodes $D(E) = \{S_1,\ldots,S_i\}$, if there is a node $S_j \in D(E)$, $1 \leqslant j \leqslant i$, such that all the nodes in $D(E) - \{S_j\}$ are neighbors of S_j, we call the relative positions of detecting nodes $S_1, \ldots, S_i$ are *legitimate*. Otherwise, the relative positions of $S_1, \ldots, S_i$ are *illegitimate*.

As each node pre-stores the neighbor information of its upstream nodes and the keys of part of its upstream nodes, it can verify not only the legitimacy of the relative positions of the detecting nodes carried in the report, but also the correctness of a MAC.

When a node receives a report R, it first checks the carried MACs as in GFFS, and then checks the legitimacy of the relative positions of all the detecting nodes according to Definition 2. If any of these nodes is not the neighbor of the CoS, the report is dropped. The pseudo-code for en-route filtering operations in NFFS is given in Algorithm 2.

Algorithm 2. En-route Filtering in NFFS

```
/* On receiving report R*/
1. Check whether or not t {$S_v$, $M_v$} tuples exist in R, $1 \leqslant v \leqslant t$; drop R otherwise.
2. Check whether or not it has not stored the neighbor information for $S_1$, drop R otherwise.
3. Check whether or not the $t - 1$ node IDs {$S_v$, $2 \leqslant v \leqslant t$} all belong to the neighbor set of $S_1$:N ($S_1$); drop R otherwise.
4. If it has one key $K \in \{K_v, 1 \leqslant v \leqslant t\}$, it computes $M = K(e)$ and see whether or not the corresponding $M_v$is the same as
   M; drop R otherwise.
5. Send R to the next hop.
```

5.2. Performance analysis of NFFS

5.2.1. Collaborative false data injection attacks

Different from existing filtering schemes, NFFS requires that each forwarding node validate not only correctness of the MACs carried in the reports, but also legitimacy of detecting nodes' relative positions. As a result, NFFS can localize the damage of compromised nodes. Let's take the nodes $S_1, \ldots, S_5$ in Fig. 3 as another example. Here we assume that $S_1, \ldots, S_5$ distribute in arbitrary areas. If the adversary abuses these nodes to fake a report R:{e; $S_1,S_2,\ldots,S_5$; $M_1,M_2,\ldots,M_5$}, then R will be dropped by the first forwarding node. Assume that the first forwarding node is S_0. If S_0 has stored the neighbor information of S_1, it then can discover that S_2 is not the neighbor of S_1; otherwise, S_0 also treats R as illegitimate for not being generated by its upstream nodes.

5.2.2. Compromise tolerance

In order to fake out false reports that NFFS cannot detect and filter out, the attacker has to capture t nodes among which there is a node S such that all other compromised nodes are its neighbor.

Theorem 3. *In NFFS, when N_c nodes are randomly captured in the network ($N_c \geqslant t$), the probability of the adversary to get at least t nodes within a $\pi \times x^2$ circular area is*

$$p(x) = \sum_{i=t}^{N_c}\left[C(N_c, i) \cdot \left(\frac{\pi x^2}{D}\right)^i \cdot \left(1 - \frac{\pi x^2}{D}\right)^{N_c - i}\right]. \tag{10}$$

Proof. We assume D_1 to be a $\pi \times x^2$ circular area. Each node falls in D_1 with probability $\pi x^2/D$, and thus the probability that exactly t nodes fall in D_1 is $p'_u = C(N_c, t) \times (\pi x^2/D)^t \times (1 - \pi x^2/D)^{Nc-t}$. Similarly, we can get the probability of the adversary to get exactly $t + 1, t + 2, \ldots,$ and N_c nodes in D_1. Therefore, the probability of the adversary to get at least t nodes within a πx^2 circular area is $p(x)$. □

Theorem 4. *In NFFS, when N_c nodes are randomly captured in the network ($N_c \geqslant t$), the probability of the adversary to get at least t nodes within a $\pi \times x^2$ circular area is*

$$p\left(\frac{r_c}{2}\right) < p_N < p(r_c). \tag{11}$$

Proof. We first note "Given $y(t \leqslant y \leqslant N_c)$ nodes, there is a node S such that the distance between any of the other $y - 1$ nodes and S is not more than r_c", "y nodes exist in a $\pi(r_c/2)^2$ circular area simultaneously", "y nodes exist in a πr_c^2 circular area simultaneously" as event a, event a_0, and event a_1, respectively. Then event a_0 implies event a, and event a implies event a_1. With Theorem 3, we get $p(r_c/2) < p_N < p(r_c)$. □

Fig. 5 illustrates the theoretical and simulation results of p_G and p_N when $t = 5, D_0/D = 0.25$ and $n = 15$. The simulation results are averaged over 10,000 random tests. From both theoretical analysis and simulation results we see that NFFS can tolerate a few more compromised nodes than GFFS. For example, when 60 nodes are compromised, the probability that the attacker completely break down the security protection of GFFS and NFFS is 0.4 and 0.05, respectively. This is because the sensing radius r_s of a node is usually larger than its transmission radius r_c. GFFS limits the damage of compromised nodes to a πr_s^2 circular area, while NFFS limits the damage of compromised nodes to an area between $\pi(r_c/2)^2$ and πr_c^2. Thus from Theorems 2 and 4 we know that the compromise tolerance ability of NFFS is a little stronger than that of GFFS.

5.2.3. Filtering effectiveness

If the adversary has compromised a large amount of nodes, it has a high probability to get t nodes within a $\pi \times r_c^2$ circular area, and thus can forge reports that cannot be detected by NFFS.

Theorem 5. *In NFFS, when a node S and $N_n - 1$ of its neighbors ($0 \leqslant N_n \leqslant t - 1$) are captured, if the adversary uses S as CoS to fake a report R, then the average number of hops R traverses before being filtered out is*

$$H_n = 1 + \sum_{d=2}^{h_0}\left(d \cdot p_d \cdot \prod_{j=1}^{d-1}(1 - p_j)\right). \tag{12}$$

Proof. Let the forwarding path from S to the sink be Path $(S) = \{S, S_1, \ldots, S_d, \text{sink}\}$, the number of S's neighbors be $|\mathrm{N}(S)|$, and the number of hops from S and S_i to sink be h_0, $h_i(1 \leqslant i \leqslant d)$, respectively. Then we get $h_i = h_0 - i$.

To fake a legitimate report R by using S as CoS, the adversary has to fake $t - N_n$ additional MACs. Because S_i stores the key for one of S's neighbors, the probability that S_i happens to have one of the corresponding $t - N_n$ keys is

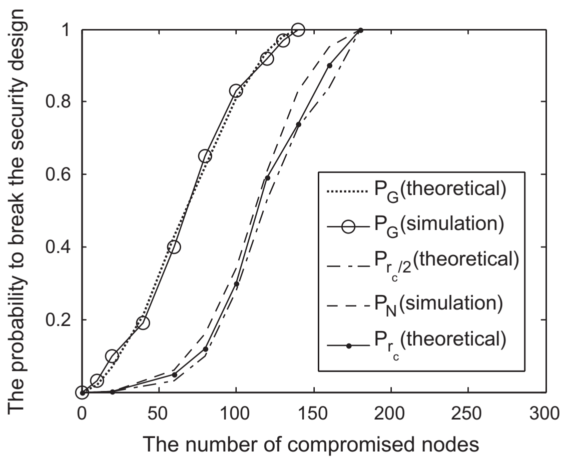

Fig. 5. Theoretical and simulation results of p_G and p_N.

$$p_i = \frac{t - N_n}{|N(S)| + 1} \cdot \frac{h_i}{h_0}. \tag{13}$$

All the intermediate nodes will check the legitimacy of the report. The probability that R passes the detection of S_1 to S_{d-1} but be detected by S_d is

$$p_d \cdot \prod_{j=1}^{d-1} (1 - p_j). \tag{14}$$

It can be easily derived that the expected number of hops is H_n. □

5.2.4. Energy consumption

The energy consumption of NFFS comes from four sources. The first is the communication overhead of each node to distribute its neighbor information and key during the pre-deployment and bootstrapping phase. The second is the communication overhead between the CoS and other detecting nodes. The third is the MAC computation overhead of the intermediate nodes. The fourth is the communication overhead of reports forwarding.

As analyzed in GFFS, we only need to consider the energy consumption of reports transmission. The size of a report in NFFS is the same as in aforementioned SEF. If the adversary uses compromised nodes from different geographical areas to fake reports collaboratively, NFFS can save energy than SEF because of its early detection and dropping of false reports. Our simulation results verified this (see Section IIV).

5.2.5. Storage overhead

NFFS requires each node to store one pre-distributed key, the neighbor information of all upstream nodes, and the keys of part of the upstream nodes. For simplicity in analysis, we assume that the network is deployed in a square region with size $50 \times 50\,\text{m}^2$ and there are totally 400 nodes with transmission range set at 2.5 m. In this scenario, the average number of neighbors of each node is 8, and the average number of paths of each node is 40. Assume that the lengths of a key and a node ID are 64 bits and 10 bits, respectively. Then each node incurs a storage overhead of about 1.2 KB, which can be met by mainstream nodes. Table 1 illustrates the comparison of mainstream filtering schemes.

6. Security analysis

In GFFS, each forwarding node verifies not only the correctness of the MACs and locations, but also the legitimacy of the locations. We consider the case that the adversary abuses the adjacent locations fetched from legitimated reports to forge a report. Although the locations included in the report are correct and legitimate, however, some of the carried MACs must be incorrect and thus will be detected by the forwarding nodes. In other words, we can simply include the location information of detecting nodes in the data report as plaintext without decreasing the security performance. Similarly in NFFS, it is also meaningless for the adversary to abuse the adjacent IDs to collude.

Table 1
Storage overhead comparison of mainstream filtering schemes.

Scheme	Storage overhead (KB)
SEF [3]	0.4
IHA [4]	4
DEFS [5]	3.5
GRSEF [6]	2.4
GFFS	0.5
NFFS	1.2

Table 2
Simulation parameters.

Parameters	Value
Interval of generating a report	2 s
Total number of reports	100
Transmission radius of sensor nodes	2.5 m
Sensing radius of sensor nodes	10 m
Power consumption of transmission	6×10^{-3} J
Power consumption of reception	1.2×10^{-3} J
Number of MACs each report carries (t)	5

7. Simulation results

We use simulations to verify our analysis. Due to space constraint, we only present results for compromise tolerance, en-route filtering and energy consumption when c is 20 and 40. We use a field size of $\pi \times 50$ m^2, where 400 nodes are uniformly distributed. One stationary sink and one stationary source sit in opposite ends of the field. The transmission time for a packet is 10 ms. For SEF and GFFS, we use a global key pool of 150 keys, which is divided into 15 partitions with 10 keys in each partition. Each node has 5 keys. Other parameters are shown in Table 2. The results are averaged over 10 simulated topologies.

In order to efficiently evaluate the filtering capabilities of related mechanisms, we introduce a new metric f, referred to as the cumulative value of the number of false reports dropped in each hop, as shown in Eq. (15). Here h and N_h means the transmission hops and the reports dropped in the hth hop, respectively. If a mechanism has good filtering capability, then each false report needs to travel only little hops before being detected. Accordingly, f $(0 \leqslant f \leqslant 100)$ won't be too small. Nevertheless, $f = 0$ means that none of the false reports can be filtered by the security mechanism.

$$f = \sum_{h=1}^{\infty}(N_h/h) \tag{15}$$

Fig. 6 illustrates how the metric f changes according to the number of compromised nodes. From Fig. 6 we have the following observations: (1) Both GFFS and NFFS outperform SEF in compromise tolerance. The number of compromised nodes that SEF,

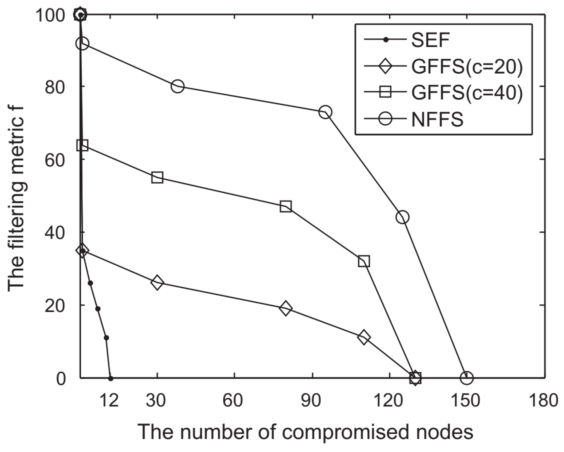

Fig. 6. The filtering metric f changes according to the number of compromised nodes.

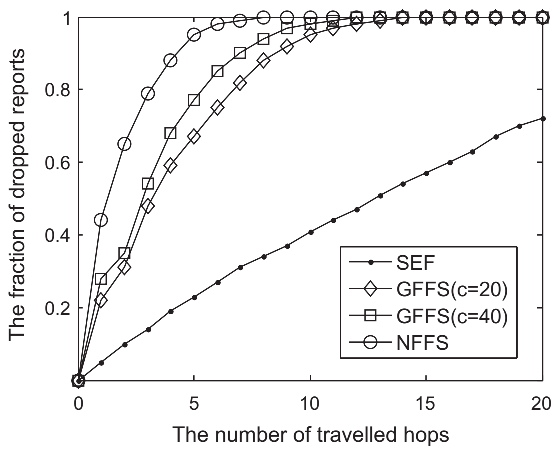

Fig. 7. Percentage of dropped false reports grows as the number of hops increases.

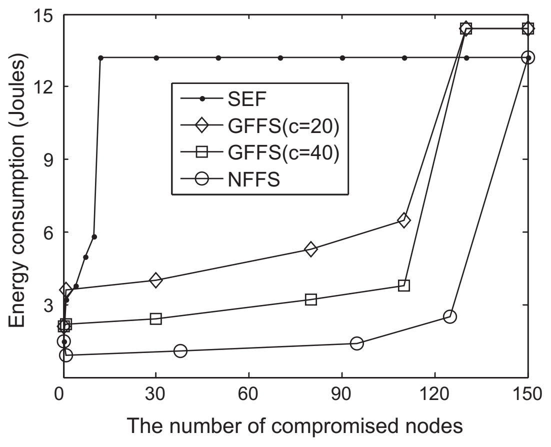

Fig. 8. The energy consumption as a function of the number of compromised nodes.

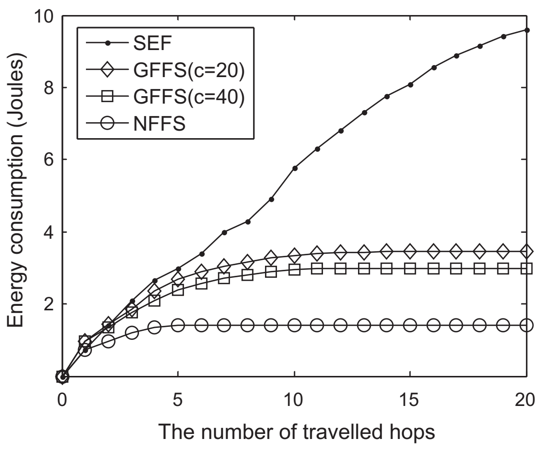

Fig. 9. The energy consumption E changes as H grows.

GFFS and NFFS can tolerate is 12, 130, and 150, respectively. (2) As N_c increases, the filtering capability of SEF decreases quickly, while the filtering capability of GFFS and NFFS decreases much more slowly. (3) With the same N_c, the filtering capability of GFFS and NFFS is stronger than that of SEF. For example, when $N_c = 12$, the value of f is 11 in SEF, 32 in GFFS ($c = 20$), 60 in GFFS ($c = 40$), and 88 in NFFS, respectively. (4) The filtering capability of GFFS increases when c increases.

Fig. 7 shows the percentage of dropped false reports as a function of traveled hops. From Fig. 7 we can see that when the number of traveled hops increases, the filtering probability of SEF, GFFS and NFFS all increase gradually. Furthermore, when the number of traveled hops is the same, the filtering performance of both GFFS and NFFS is better than that of SEF. For example, when $H = 5$, the fraction of dropped false reports is 23% in SEF, 67% in GFFS ($c = 20$), 77% in GFFS ($c = 40$), and 95% in NFFS, respectively.

Fig. 8 plots how the energy consumption E changes according to N_c. From Fig. 8 we can see that, when $c = 20$, only when $N_c < 4$, GFFS consumes little more energy than SEF. In other cases, both GFFS and NFFS consume much less energy than SEF. For example, when $N_c = 30$, the energy consumption is 13.2 J in SEF, 4 J in GFFS ($c = 20$), 2.4 J in GFFS ($c = 40$), and 1.8 J in NFFS, respectively. This is because the filtering probability of GFFS is the same as SEF in the situation of $c = 20$ and $N_c < 4$, but the packet length of GFFS is larger than SEF. While in other situations, GFFS and NFFS consume less energy than SEF.

Fig. 9 shows how the energy consumption E changes when the number of hops H grows. From Fig. 9 we can see that only when $H \leqslant 2$, GFFS consumes a little more energy than SEF. While in other cases, both GFFS and NFFS can consumes less energy than SEF because of their early filtering of false reports.

8. Conclusion

False data filtering is an important issue in wireless sensor networks. In this paper, we considered a new type of false data injection attacks called collaborative false data injection, and proposed two schemes to defend such attacks. In collaborative false data injection attacks, multiple compromised nodes collaboratively forge a fake report and inject the report into the network. This type of attacks is hard to defend with existing approaches, because they only verify a fixed number of message authentication codes (MACs) carried in the data report but the adversary can easily obtain enough compromised nodes from different geographical areas of the network to break their security. Our novel solution is to bind the keys of sensor nodes to their geographical locations, and verify the legitimacy of a data report by checking whether the locations of the sensors endorsing the report are logical (e.g., the sensors should be close enough to each other to sense the same event). We proposed two filtering schemes: The geographical information based false data filtering scheme (GFFS) which utilizes the absolute positions of sensors in the verification, and the neighbor information based false data filtering scheme (NFFS) which utilizes relative positions of sensors when absolute positions cannot be obtained. Analysis and simulation results demonstrate that both GFFS and NFFS are able to resist the collaborative false data injection attacks effectively and thus can tolerate much more compromised nodes than existing schemes. As for future work, we plan to extend our research results to sensor networks with mobile sink or multiple sinks.

Acknowledgment

This work is sponsored by the National Natural Science Foundation of China under Grant Nos. 61232001, 61103203, 61173169.

References

[1] E. Ayday, F. Delgosha, F. Fekri, Location-aware security services for wireless sensor networks using network coding, IEEE Conference on Computer Communications, 2007, pp. 1226-1234.
[2] R. Blom, An optimal class of symmetric key generation systems, in: Advances in Cryptoloty, Proceedings of EUROCRYPT 84, Lecture Notes in Computer Science, 1985, pp. 335–338.
[3] B. Bloom, Space/time trade-offs in hash coding with allowable errors, Communications of the ACM 13 (7) (1970) 422–426.
[4] A.K. Bashir, S.J. Lim, C.S. Hussain, M.S. Park, Energy efficient in-network RFID data filtering scheme in wireless sensor networks, IEEE Sensors Journal 11 (2011) 7004–7021.
[5] P. Bose, B. Morin, I. Stojmenovic, J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, Wireless Networks 7 (6) (2011) 609–616.
[6] W. Du, J. Deng, Y. Han, P. Varshney, A pairwise key pre-distribution scheme for wireless sensor networks, in: Proceedings of the ACM CCS, 2004, pp. 228–258.
[7] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, Resilient network coding in the presence of byzantine adversaries, Proceedings of the IEEE Transactions on Information Theory 54 (6) (2008) 2596–2603.
[8] W.K. Lai, C.S. Fan, L.Y. Lin, Arranging cluster sizes and transmission ranges for wireless sensor networks, Information Sciences 183 (1) (2012) 117–131.
[9] R.X. Lu, X.D. Lin, H.J. Zhu, X.H. Liang, X.M. Shen, BECAN: a bandwidth-efficient cooperative authentication scheme for filtering injected false data in wireless sensor networks, IEEE Transactions on Parallel and Distributed Systems 23 (1) (2012) 32–43.
[10] M. Ma, Resilience of sink filtering scheme in wireless sensor networks, Computer Communications 30 (1) (2006) 55–65.
[11] G. Mao, B. Fidan, B. Anderson, Wireless sensor network localization techniques, Computer Networks: The International Journal of Computer and Telecommunications Networking (2007) 2529–2553.
[12] K. Naresh, K.P. PrDadeep, K.S. Sathish, An active en-route filtering scheme for information reporting in wireless sensor networks, IJCSIT: International Journal of Computer Science and Information Technologies 2 (4) (2011) 1812–1819.
[13] A. Parakh, S. Kak, Space efficient secret sharing for implicit data security, Information Sciences 181 (2) (2011) 335–341.
[14] F.Y. Ren, H.N. Huang, C. Lin, Wireless sensor networks, Journal of Software 14 (7) (2003) 1282–1291.
[15] K. Ren, W. Lou, Y. Zhang, Providing location-aware end-to-end data security in wireless sensor networks, in: Proceedings of the IEEE Conference on Computing and Communicating, 2006, pp. 585–598.
[16] Z. Su, C. Lin, F.J. Feng, F.Y. Ren, Key management schemes and protocols for wireless sensor networks, Journal of Software 18 (5) (2007) 1218–1231.
[17] M. Saleem, I. Ullah, M. Farooq, BeeSensor: an energy-efficient and scalable routing protocol for wireless sensor networks, Information Sciences 200 (19) (2012) 38–56.
[18] H. Wang, Q. Li, PDF: a public-key based false data filtering scheme in sensor networks, in: Proceedings of the International Conference on Wireless Algorithms, Systems and Applications, 2007, pp. 129–138.

[19] Z. Yu, Y. Guan, A dynamic en-route scheme for filtering false data injection in wireless sensor networks, in: Proceedings of 25th Annual Joint Conference of the IEEE Computer and Communications Societies, 2006, pp. 1–12.
[20] H. Yang, S. Lu, Commutative cipher based en-route filtering in wireless sensor networks, in: Vehicular Technology Conference, 2004, pp. 1223–1227.
[21] L. Yu, J.Z. Li, Grouping-based resilient statistical en-route filtering for sensor networks, in: Proceedings of 28th Annual Joint Conference of the IEEE Computer and Communications Societies, 2009, pp. 1782–1790.
[22] X.Y. Yang, J. Lin, P. Moulema, W. Yu, X.W. Fu, W. Zhao. A novel en-route filtering scheme against false data injection attacks in cyber-physical networked systems, in: Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS), 2012, pp. 92–101.
[23] F. Ye, H. Luo, L. Zhang, Statistical en-route filtering of injected false data in sensor networks, in: Proceedings of 23th Annual Joint Conference of the IEEE Computer and Communications Societies, 2004, pp. 2446–2457.
[24] B. Yu, M. Yang, Z. Wang, C.S. Gao, Identify abnormal packet loss in selective forwarding attacks, Chinese Journal of Computers 29 (9) (2006) 1542–1552.
[25] H. Yang, F. Ye, Y. Yuan, S. Lu, W. Arbaugh, Toward resilient security in wireless sensor networks, in: Proceedings of the 6th ACM International Symposium on Mobile Ad hoc Networking and Computing, 2005, pp. 34–45.
[26] F. Yang, X.H. Zhou, Q.Y. Zhang, Multi-dimensional resilient statistical en-route filtering in wireless sensor networks, LNCS: Lecture Notes in Computer Science (2010) 130–139.
[27] W.S. Zhang, G.H. Cao, Group rekeying for filtering false data in sensor networks, in: Proceedings of the IEEE Conference on Computing and Communicating, 2005, pp. 503-514.
[28] L. Zhou, C. Ravishankar, A fault localized scheme for false report filtering in sensor networks, in: Proceedings of the IEEE International Conference on Pervasive Services, 2005, pp. 59–68.
[29] S. Zhu, S. Setia, S. Jajodia, An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks, in: Proceeding IEEE symposium on Security and privacy, 2004, pp. 259–271.
[30] Y. Zhang, J. Yang, H. Vu, The interleaved authentication for filtering false reports in multipath routing based sensor networks, in: Proceedings of 20th International Parallel and Distributed Processing Symposium, 2006, pp. 1–10.