An active detection of compromised nodes based on en-route trap in wireless sensor network

Jiang-Tao Wang and Zhi-Xiong Liu

Abstract
With the development and wide use of wireless sensor network, security arises as an essential issue since sensors with restrict resources are deployed in wild areas in an unattended manner. Most of current en-route filtering schemes could filter false data effectively; however, the compromised nodes could take use of the filtering scheme to launch Fictitious False data Dropping attack, the detection of this attack is extremely difficult since the previous hop node is unable to distinguish whether the forwarding node dropped a false data report with incorrect Message Authentication Codes or a legitimate report. This is the first attempt to address the Fictitious False data Dropping attack; in this article, we propose an Active Detection of compromised nodes based on En-route Trap to trap compromised nodes in the scenario of a false data dropping. A trust model is used to evaluate trust level of neighbor nodes with respect to their authentication behaviors. A detecting algorithm of compromised node is used to detect compromised nodes. Simulation results showed that our scheme can address the Fictitious False data Dropping attack and detect 60% of compromised nodes with a low false positive rate; consequently, the packet accuracy of an Active Detection of compromised nodes based on En-route Trap increases rapidly and reaches to 86%.

Keywords
Compromised node detection, fictitious false data dropping attack, en-route trap

Date received: 24 April 2021; accepted: 24 July 2021
Handling Editor: Lyudmila Mihaylova

Introduction
Wireless sensor networks (WSNs) are widely used to monitor environments in many applications such as forest fire monitoring or military surveillance, it is a common view that WSN will play a vital role in Internet of Things (IOT) or next generation network. However, in most cases, sensor nodes in WSN are deployed in remote wild areas or even hostile environments in an unattended manner, with restrict limited battery power, communication capability, and computation capability.1,2 The adversary may capture some sensor nodes physically, acquire the secret information stored in these nodes, and take full control of these compromised nodes. Even worse, these compromised nodes are leveraged by the adversary to launch many kinds of insider attacks such as false data injection attack,3 data dropping attack, or selective forwarding attack. Without identifying compromised nodes, they may continuously attack sensor system and waste precious limited resources of the network. Compromised nodes detection is of great importance for ensuring the security of WSN.

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
However, as we discussed in next related works section, most proposed filtering schemes could filter false data injected by compromised nodes, but these filtering schemes cannot detect compromised nodes. Similarly, most proposed compromised node detection (CND) mechanisms are not technically mature to be applied into WSN. Moreover, the filtering scheme itself may become the attack target of compromised nodes. In Statistical En-route Filtering (SEF), the compromised nodes could take use of the filtering scheme to launch Fictitious False data Dropping (FFD) attack, the FFD attack is described as a compromised relay node on route drop a legitimate data report and declares it is a false data. The detection of this FFD attack is difficult, since the monitor node may lack of the key to distinguish it is a normal false data dropping or an FFD attack of compromised nodes? To our knowledge, the FFD attack of compromised nodes is not disused in any existing literatures, it is obviously a security hole of existing en-route filtering scheme, and it motivated us to design a scheme to address this security issue as described in this article.

In this article, we focus on addressing this FFD attack, and other attack forms of compromised nodes are beyond the scope of this article. Based on the analysis of FFD attack, we found that the false data dropping scenario provides us with a perfect opportunity to trap compromised nodes, and this led us to design an Active Detection of compromised nodes based on En-route Trap (ADET). In ADET, all compromised nodes possessing indicated key within one-hop range are trapped where a false data dropping is reported, and a trust model is incorporated in the en-route trap to differentiate normal nodes and compromised nodes.

In this article, the data transferring and false data filtering scheme of our ADET are developed based on SEF; the motivation behind our design relying on SEF is an initial study to filtering false data. After that, many other research works based on SEF such as Grouping-based Resilient Statistical En-route Filtering (GRSEF) scheme and Location-Based Resilient Security (LBRS) solution are proposed and described in Yu and Li and Yang et al. In the experiment section, we compared our ADET with SEF in aspects such as packet accuracy and energy efficiency.

The main contributions of this work are listed as follows:

- An active en-route trap scheme is designed to trap compromised nodes in case of a false data dropping.
- A trust model is used to evaluate trust level of neighbor nodes with respect to their authentication behaviors.
- A detecting algorithm of compromised node is used to detect compromised nodes.

**Related works**

In recent years, many research works which focus on addressing the security issues caused by compromised nodes are proposed and described as follows.

Once the sensor nodes are compromised, they can launch false data injection attack. Thus, several en-route filtering schemes have been proposed to drop the false data en-route before they reach the sink. These filtering schemes could filter false data efficiently; however, they are not able to detect compromised nodes.

In recent years, some CND schemes are proposed as following. Ye et al. propose a probabilistic nested marking scheme to locate colluding compromised nodes in false data injection attacks. Zhang et al. propose the COmpromised nOde Locator (COOL) system which is an intrusion detection system that detects the compromised nodes using the relationship between incoming and outgoing messages. In Xu et al., CND scheme is proposed to detect compromised nodes in WSN; it uses common application features and adjusts detection behavior when there are no periodic transmissions or lack of communications between nodes networks. Moreover, several software-based attestation schemes for node compromise detection in sensor networks also have been proposed. However, they are not readily applied into regular sensor networks due to several limitations. Yang et al. present two distributed schemes toward making software-based attestation more practical; neighbors of a suspicious node collaborate in the attestation process to make a joint decision. In Lin, a Couple-based node compromise detection (CAT) is proposed to early detect compromised nodes using node couples; it is the first attempt to detect compromised node in the first stage. In Al-Riyami et al., an Adaptive Early Node Compromise Detection Scheme (AdaptENCD) for Hierarchical WSNs is proposed to early detect compromised nodes; this scheme is an enhanced version of CAT. To achieve a low false positive ratio in the presence of various levels of message loss ratios, two ideas are used in the design. The first is to use cluster-based collective decision-making to detect node compromises. The second is to dynamically adjust the rate of notification message transmissions in response to the message loss ratio in the sender’s neighborhood. However, both CAT and AdaptENCD are designed to detect compromised nodes in the first stage; they are not designed to defend attacks of compromised nodes in the data transferring stage. In Iqbal et al., a weighted fusion scheme is proposed to locate and disregard the information from compromised sensors in a WSN; however, it does not figure out how to detect compromised nodes. In Neggazi et al., a novel silent self-stabilizing algorithm for computing a minimal edge-monitoring set in sensor network is proposed; monitoring nodes can detect any
malicious actions such as delaying, dropping, modifying, or even fabricated packets, but the details of the detection of malicious actions are not mentioned in the article. In Remesh Babu Raman, a hybrid double layered security strategy for sensed data is proposed: the first step of security is applied by appending a Keyed Message Authentication Code (HMAC) to the sensed data by Secure Hash Algorithm (SHA-2/512), and the second step of security is implemented by a modified form of Constrained Random Perturbation-based Pairwise keyY (CARPY+) mechanism; however, the communication between neighbor nodes is based on a pairwise key channel which is not energy efficient. In Zhang et al., an application independent framework for accurately identifying compromised sensor nodes is proposed. The framework provides an appropriate abstraction of application-specific detection mechanisms and models the unique properties of sensor networks. Based on the framework, an alert reasoning algorithm is used to identify compromised nodes. In Ho et al., a zone-based node compromise detection and revocation scheme in WSNs is proposed. The main idea behind this scheme is to use sequential hypothesis testing to detect suspect regions in which compromised nodes are likely placed. In these suspect regions, the network operator performs software attestation against sensor nodes, leading to the detection and revocation of the compromised nodes.

All above proposed schemes aim to detect compromised nodes in WSN. However, some schemes have corresponding limitations or based on assumptions, and some schemes involve too much extra communication which means they introduce unnecessary overhead. More important is that all detection schemes as mentioned above belong to passive mode; this motivates us to introduce our ADET which is an active scheme to trap compromised nodes.

### Foundation technologies

#### SEF

In this section, we briefly describe the two foundation technologies that we borrowed to design our ADET, which are classical SEF scheme and Pervasive Trust Model (PTM).

In our design, we build the proposed ADET scheme on the framework of statistical en-route filtering of injected false data (SEF). In SEF, a global key pool is divided into $n$ ($n > T$) partitions each of which containing $m$ keys. Every node randomly picks $k$ keys from one partition and the nodes holding keys from the same partition form a group. Each key is composed of a key index and a key value. The key sharing probability $q = k/m$ is the probability of any two nodes from the same group sharing a common key. Furthermore, SEF is independent of data dissemination protocols (Figure 1).

Once a stimulus is detected by multiple sensors, each of the detecting sensors reports its sensed signal density and one of them is elected as the Center-of-Stimulus (CoS). The CoS collects and summarizes all the received detection results, and produces a synthesized report on behalf of the group. The report which is attached with $T$ Message Authentication Codes (MACs) is then forwarded toward the sink. The form of final data report is described as follows

$$\{R_n, L, t, E, i_1, M_{i_1}, i_2, M_{i_2}, \ldots, i_T, M_{i_T}\}$$

where $i_j$ is the index of the key, and $M_i$ is the MAC which is computed by corresponding key.

The filtering algorithm in SEF as described in Figure 2 is used to filter the false data report.

#### Trust model

In this section, we briefly describe the trust model of PTM which is used in our ADET to differentiate normal nodes and compromised nodes in case of a false data dropping.

This PTM has the feature of low communication, computation, and storage overhead. Furthermore, the trust evaluation of PTM also has the feature of decrease-fast-and-increase-slow which is suitable for our ADET. It is used in our ADET to evaluate the trust levels of neighbor nodes with respect to their behaviors. However, the recommendation trust is not considered for overhead consideration. The following formula is used to calculate new trust evaluation $R(A, C)_\text{new}$

$$V_a = \left(1 - \frac{A_n}{\text{Total}_a}\right)^*p_0(a)$$

$$R(A, C)_\text{new} = \begin{cases} V_a*\beta + R(A, C)_\text{old}*(1 - \beta) & V_a > 0 \\ 0 & \text{else} \end{cases}$$

(1)
where the range of $R(A, C)_{new}$ is $[0,1]$, $Total_a$ is the total number of interactions between node $A$ and node $C$, $AN$ is the total number of negative interactions, $\omega$ is the interaction weight of behavior which is set to 1 in case of positive behavior and set to 0.5 in case of negative behavior, and $\beta$ is the embodied history interaction weight.

**System models and design goal**

In this section, we formulate the network model, the threat model, and identify the design goal.

**Network model**

Many sensor nodes are deployed on a vast geographic area. The sensor nodes are randomly distributed and dense enough so that each stimulus can be detected by multiple nodes. Each node knows its location with a localization system or Global Positioning System (GPS). Each node is assigned with a unique $ID$ and $k$ keys before deployment as described in SEF. Both the $ID$ and the keys are known to the sink. After deployment, each node acquires the $k$ key indexes from each of its one-hop neighbor nodes. When an event is detected by multiple nodes, the CoS produces a synthesized data report on behalf of the group. The data report which is attached with $T$ MACs and $T$ key indexes is then forwarded toward the sink with Geographic and Energy Aware Routing (GEAR) algorithm. The SEF scheme is used to filter false data en-route; some false data reports will be filtered out en-route while others may reach the sink without being detected. Each forwarding node will store a copy of the data report for a system predefined time $T_c$ for authentication purpose in en-route trap. Whenever a false data report is detected by a relay node, it will drop the data report and report a false data dropped message to its previous hop node, namely, monitor node.

The sink is a data collection center with sufficient computation and storage capabilities, and it is equipped with tamper-resistant hardware. It serves as the final goal-keeper for the system. When it receives an event report, it can verify all the MACs carried in the report because it has complete knowledge of the global key pool, and all false data reports with incorrect MACs that sneak through en-route filtering will then be detected.

**Threat model**

We assume that the adversary can compromise multiple sensor nodes, obtain the security information embedded in these nodes, and take full control of them. However, the attacker cannot compromise the sink which is well-secured.

In this article, we focus on addressing the FFD attack; the en-route trap scheme is designed to trap compromised nodes in case of false data dropping. Other attack patterns of compromised nodes such as report tampering attack or collusion attack which means the monitor node may also be compromised are beyond the scope of this article, and we leave them to future works.

We also assume the compromised node adopt an “on and off” pattern to launch attack; a compromised node has the probability of $\alpha$ to launch FFD attack whenever it is involved in a forwarding interaction, where $\alpha$ is a system predefined parameter with the range of $[0–1]$.

**Design goal**

For a sensor network with above settings and models, our design goal is to detect compromised nodes which launching the FFD attack and exclude them from the network. The proposed scheme meets the following requirements:

- The scheme can address FFD attack at relay nodes and simultaneously filter false data on route.

---

**Figure 2.** Filtering algorithm of false data report in SEF.
The scheme can detect compromised nodes with a low false positive rate. The scheme introduces low extra energy overhead and storage overhead to existing routing algorithm.

The main goal of our design is to detect compromised node with low false positive rate which launching the FFD attack; other forms of attack from compromised nodes are beyond the scope of this article. To avoid unnecessary energy consumption and prolonged network life, the communications needed to detect the FFD attack and confirm the compromised node should be strictly limited to local one-hop range.

ADET

System architecture

The ADET is designed to building a pure distributed compromised nodes detection system with low energy overhead; each node runs ADET independently and continuously monitoring data report forwarding of neighbor nodes, recording positive or negative behavior of corresponding nodes in case of false data dropping. With the accumulation of negative behaviors, once the trust evaluation falls below a system predefined threshold, a detection algorithm of compromised node is used to detect compromised node. Each node made its own decision relying on its own observations independently. The work process of ADET is described in Figure 3.

En-route trap module. The en-route trap module is responsible for authenticating the legitimacy of the false data dropping and actively trapping compromised nodes. The formally definition of FFD attack is described as follows.

Definition of FFD attack. An FFD attack is defined as a legitimate data report is drop by next relay node, and the relay node declares it drop a false data report with an incorrect MAC.

In case of an FFD attack, once after $N_a$ forwarded the data report to $N_b$, it is monitored that the data report is not forwarded by $N_b$ within a predefined period $T_c$, instead it received a false data drop message from $N_b$ with the content of $\{R_n, \text{keyindex}, \text{MAC}\}$, announcing the drop data report is a false data report, indicating corresponding key index and MAC does not match. First $N_a$ will check the existence of $\{R_n, \text{keyindex}, \text{MAC}\}$ by compare it with the copied data report (notice that $R_n$ is data report serial number, and each data report will be copied and stored at the relay node for a predefined period), then $N_a$ found it cannot distinguish it is a legitimate false data dropping or an FFD attack of $N_b$, since it do not possess the indicated key to verify the MAC (the possibility of $N_a$ possesses the indicated key is negligible). In fact, our ADET can not directly detect FFD attack; however, this false data dropping scenario provides us with a perfect opportunity to trap compromised nodes, in this way the FFD attack is indirectly addressed as we described in our en-route trap scheme. The details of work processes in en-route trap are described in section “En-route trap.”

Trust module. The trust module is responsible for evaluating the trust level of neighbor nodes relying on the inputs from en-route trap module. The PTM trust model which is used in our ADET has the feature of decrease-fast-and-increase-slow, and it means a negative behavior’s weight is bigger than a positive behavior. With the accumulation of negative behaviors, the suspicious compromised node’s trust evaluation will decrease rapidly, once it falls below a system predefined threshold value $\theta$, a detection algorithm of compromised node will be activated to detect compromised nodes in CND module. The details of how the trust model is used to differentiate normal nodes and compromised nodes are described in section “En-route trap.”

CND module. The CND module is responsible for detecting the compromised nodes. In our ADET, a detection algorithm of compromised node is used to detect a compromised node, we believe the detection of a compromised node need joint work of neighbor nodes, otherwise the slander attacking to each other between normal nodes and compromised nodes will introduce unnecessary false positive rate.
Similarly, we believe the compromised nodes do not have the motive for taking use of this detection algorithm to directly attack normal nodes. The reason relies on that, if the normal nodes are falsely detected and isolated from network, these compromised nodes will lose the targets of attack, and even the compromised nodes themselves may be isolated indirectly. In fact, a smart compromised node prefers to launch attack by taking use of neighboring normal nodes, just as the attack pattern of what we described in the FFD attack, this is also the motive we proposed ADET. The details of the CND algorithm are described in Figure 4.

En-route trap

**Description of en-route trap.** The en-route trap scheme is designed to trap compromised nodes in case of a false data dropping. When $N_a$ received a false data drop message from next relay node $N_b$ with the content of $\{R_n, keyindex, MAC\}$, announcing that the drop data report is a false data, indicating corresponding key index and $MAC$ does not match. First $N_a$ will check the authenticity of $\{R_n, keyindex, MAC\}$ by compare it with the copied data report, then $N_a$ found it is unable to distinguish it is a legitimate false data dropping or an FFD attack of $N_b$ since the possibility of $N_a$ possessing indicated key to verify the $MAC$ is negligible. However, as we mentioned in the network model, each node acquires the $k$ key indexes from each of its one-hop neighbors at the beginning phase after deployment, it means neighbors of $N_a$ may possess the indicated key to verify the $MAC$. $N_a$ can ask those neighbors which possess the indicated key to help him making the judgment. A question message with content of $\{ID_Na, LE, t, E, keyindex, MAC\}$ which denotes “is this an incorrect $MAC$?” will be broadcasted by $N_a$ in one-hop range and asking for replies. Notice that the $\{LE, t, E\}$ is derived from copy of the data report. Those neighbors who possess the indicated key could not refuse the request since $N_a$ knows who possess the indicated key. All replies will be sent back to $N_a$. The choices left to these neighbors are to reply to the question honestly or dishonestly for slander attack purpose.

After $N_a$ received all replies from these neighbor nodes, it still cannot distinguish which node replies to the question honestly and which node lies, since $N_a$ do not possess the indicated key, it also means the legitimacy of the false data dropping could not be verified. In fact, our en-route trap scheme cannot directly detect the FFD attack; however, the false data dropping scenario provides us with an opportunity to trap compromised nodes, and all compromised nodes possessing indicated key within one-hop range where a false data dropping is trapped in our en-route trap scheme.

If all replies of corresponding neighbor nodes fall into one group, then $N_a$ will take it as a legitimate false data dropping since it is unable to deny it. Otherwise, the different replies from neighbor nodes break the tip of the iceberg. Here, $N_a$ can act as an independent judge to group these nodes, those nodes who reply with “Yes, it’s an incorrect MAC” fall into one group while other nodes who reply with “No, it’s not an incorrect MAC” fall into another group, notice that we can treat $N_b$ as it has already replied with “Yes, it’s an incorrect MAC.” After grouping all the replies, $N_a$ will broadcast the grouping result. Obviously, there exists only one true answer; thus, a node in one group can treat a node in the other group as a suspicious compromised node since it afforded an opposite reply, then each node will record a negative behavior on each neighbor node in the other group, while each node will record a positive behavior on each neighbor node in the same group, all behaviors will be recorded in our trust model. In this way, the normal nodes and compromised nodes are
differentiated; the real compromised nodes are trapped by $N_a$, who act as an independent judger.

The scenarios of a legitimate false data dropping and an FFD attack as described in Figures 5 and 6 are used to interpret how our en-route trap scheme works.

**Legitimate false data dropping scenario.** In Figure 5, a normal node $N_b$ reports a legitimate false data dropping to $N_a$, indicating corresponding key index and $MAC$ does not match. Suppose four neighbor nodes (besides $N_b$) of $N_a$ possess the indicated key, and was forced to reply to the question “is this an incorrect MAC?,” two normal nodes will use the indicated key to verify the $MAC$ and reply with “YES,” while the other two compromised nodes dishonestly reply with “NO” to slander attack $N_b$. The compromised nodes and normal nodes are grouped into two different groups by opposite replies, each node will record a negative behavior on each neighbor node in the other group, while each node will record a positive behavior on each neighbor node in the same group, in this way normal nodes and compromised nodes are differentiated and the real compromised nodes are trapped by $N_a$.

Of course, these two compromised nodes can also choose to honestly reply with “YES” to conceal themselves, then all nodes including $N_b$ reply with “YES,” there exists only one group and it seems the en-route trap does not work. However, it provides us with an opportunity to trap compromised nodes.

**FFD attack scenario.** In Figure 6, a compromised node $N_b$ dropt a legitimate data report and send a false data dropt message to $N_a$, indicating corresponding key index and $MAC$ does not match. Two normal nodes reply with “NO,” while the other two compromised nodes dishonestly reply with “YES” to help concealing the compromised node $N_b$. The compromised nodes and normal nodes are grouped into two different groups by opposite replies, and each node will record a negative behavior on each neighbor node in the other group, while each node will record a positive behavior on each neighbor node in the same group. In this way normal nodes and compromised nodes are differentiated and the real compromised nodes are trapped.

Of course, these two compromised nodes can also choose to honestly reply with “NO” to conceal themselves, then, at least $N_b$ replies with “YES,” there exist still two groups, and it make sense that at least the two normal nodes and $N_b$ will record a negative behavior to each other. It seems the en-route trap still works in this scenario.

**Collusion attack scenario.** Another scenario is when a compromised node is forced to verify an $MAC$ from an FFD attack, the compromised node knows it is not an incorrect $MAC$, it may choose to honestly reply with “NO” to conceal itself. At the same time, it may also have the motive to dishonestly reply with “YES” to help concealing the compromised node $N_b$, and hope no one could find out; however, if another normal node replies with “NO,” then both will be recorded a negative behavior by the normal node. This scenario provides us with a perfect opportunity to trap en-route collude nodes, and it partially detected collusion attack in some extent.

**Summary of en-route trap.** In summary, our en-route trap scheme cannot directly detect FFD attack. However, it may still work to trap compromised nodes in most scenarios of false data dropping. No matter how the compromised nodes choose to reply, honestly or dishonestly? The en-route trap scheme can provide us with a perfect opportunity to trap compromised nodes. Nodes from one group will treat nodes in the other group as suspicious compromised nodes, and with the accumulation of negative behaviors, normal nodes and compromised nodes are differentiated.

In the simulation experiment, the compromised nodes are set to have the probability of 50% to reply to the question honestly and the other 50% dishonestly. The attacking to each other of nodes in different groups will led to an interesting phenomenon which is named by “shutter island” in this article; we will explain this in section “Shutter island.”
The work process of en-route trap is described in Figure 7.

Performance evaluation of en-route trap

In this section, we briefly quantify the detection power of our ADET in an FFD attack scenario. Once a compromised node launched an FFD attack as showed in Figure 8, at least the compromised node \( N_b \) replies with “YES,” if there exist any normal neighbor nodes of \( N_a \) which possess indicated key, these normal nodes are supposed to verify the \( MAC \) and honestly reply with “NO;” thus, two groups exist in the en-route trap scheme. Note that a successful detection of FFD attack is defined as the compromised node which launch this attack is recorded a negative behavior by any normal nodes.

Suppose \( N_{cab} \) denotes the set of common neighbor nodes of \( N_a \) and \( N_b \). According to our en-route trap, each node in one group will record a negative behavior to each neighbor node in the other group, it means our en-route trap works if any normal nodes possessing indicated key exist in \( N_{cab} \). Thus, the probability of successful detection of FFD attack can be denoted as

\[
P_s = P(X > 0)
\]

where \( X \) denotes the number of normal nodes possessing indicated key in \( N_{cab} \).

Theorem. Suppose each key is uniformly assigned to all nodes before deployment, and the distribution of nodes follows the Poisson distribution in a circle sensor network, then the probability of successful detection of FFD attack can be denoted as

\[
e^{-N_n \frac{k}{nm}} \frac{3r^2}{\sqrt{2\pi \sqrt{2}}} < P_s < e^{-N_n \frac{k}{nm} R^2}
\]

where \( N_n \) denotes the total number of normal nodes in sensor network, \( k \) denotes the number of keys possess by each node, \( n \) denotes the number of partitions in global key pool, \( m \) denotes the number of keys in each partition, \( r \) denotes the communication range of node, and \( R \) denotes the radius of sensor network.

Proof. As we supposed, each key is uniformly assigned to all nodes before deployment; thus, the total number of normal nodes which possess the indicated key can be denoted as

\[
N_{ek} = N_n \frac{k}{nm}
\]

Figure 9 shows the upper and lower limits of the common area of two neighbor nodes. Let \( E(X) \) denotes the mathematical expectation of \( X \), then the upper and lower limits of \( E(X) \) can be denoted as

\[
N_n \frac{k}{nm} \frac{3r^2}{\pi R^2} < E(X) < N_n \frac{k}{nm} R^2
\]
As we supposed the distribution of nodes follows the Poisson distribution, thus the upper and lower limits of $P_s$ can be denoted as

$$P_s = P(X > 0) = 1 - P(X = 0)$$

$$e^{-N_{nm} \frac{\hat{m} \sigma^2}{2R^2}} \times \left(-N_{nm} \frac{k \sigma^2}{nm R^2}\right)^0$$

$$\Rightarrow 1 - \frac{e^{-N_{nm} \frac{\hat{m} \sigma^2}{2R^2}} \times \left(-N_{nm} \frac{k \sigma^2}{nm R^2}\right)^0}{0!}$$

$$< P_s < 1 - \frac{e^{-N_{nm} \frac{\hat{m} \sigma^2}{2R^2}} \times \left(-N_{nm} \frac{k \sigma^2}{nm R^2}\right)^0}{0!}$$

$$\Rightarrow 1 - e^{-N_{nm} \frac{\hat{m} \sigma^2}{2R^2}} < P_s < 1 - e^{-N_{nm} \frac{k \sigma^2}{nm R^2}}$$

The performance evaluation of detection power of en-route trap will be simulated in the simulation experiment.

**Shutter island**

According to our en-route trap scheme, compromised nodes and normal nodes always prefer to slander attack each other, and with the work of en-route trap, normal nodes and compromised nodes are differentiated. However, normal nodes could compensate their trust evaluation of each other through positive behaviors which is defined by replying to the question honestly, while compromised nodes cannot compensate their trust evaluation since they adopt a “on and off” strategy to reply to the question honestly or dishonestly. With the accumulation of negative behaviors and the work of our trust model, compromised nodes will be detected and isolated gradually. In some local areas, the relative shorter routes to sink are cut off due to the isolation of compromised nodes, notice that GEAR routing algorithm is adopted in the experiment of our ADET; thus, latter data reports must search another route to bypass these areas. The routing price of nodes within these areas is increased, and it also means nodes within these areas have less opportunity to participate forwarding data reports. These areas together with areas at edge of sensor network form shutter islands. In some extent, shutter islands protected rest undetected compromised nodes within it from being detected, since nodes within shutter islands have less opportunity to participate forwarding data reports. Our simulation experiment results testified to the existence of shutter islands.

**ADET scheme overview**

The proposed ADET scheme follows the general en-route filtering framework as described in SEF. It consists of five phases. In the following content of this section, we briefly introduce these phases.

**Pre-deployment phase.** Before deployment, each node is assigned with a unique $ID$, each node randomly chooses one of the $n$ partitions to join in and randomly obtain $k$ authentication keys from $m$ keys in that partition; the key assignment is designed carefully so that each key is uniformly assigned to all nodes.

**Bootstrapping phase.** During the bootstrapping phase, each sensor node broadcasts a hello message to its neighbors within communication range. Once received the ACK message, the neighbor relationship between two nodes is established. Each node acquires the $k$ key indexes from each of its neighbor nodes. After that, each node assigns an initial trust evaluation to each of its one-hop neighbors, for simplicity, the initial trust evaluation $R(A,C)_{initial}$ is set to 1 which denotes the belief that the behaviors of all nodes are positive at the bootstrapping phase, accordingly, the initial number of positive and negative behaviors is set to 1.

**Robust report endorsement phase.** When an event occurs, all detecting nodes are organized into a cluster and reach an agreement on the event $E$, event report is consisting of $E$, event location $L_E$, and event time $t$. After the CoS election process finishes, a detecting node randomly selects one of its keys, and generates an $MAC$ as follows

$$M_i = MAC(K_i, L_E \| t \| E)$$

where $\|$ denotes the stream concatenation and $MAC(a, b)$ computes the $MAC$ of message $b$ using key $a$; many crypto-graphic one-way functions may serve this purpose. Each detecting node then sends $\{i, M_i\}$ to the CoS node. The CoS node collects all $\{i, M_i\}$s from detecting nodes and select $T$ $\{i, M_i\}$ tuples from $T$ different partitions. The final report sent out by CoS to the sink looks like

$$\{R_n, L_E, t, E, i_1, M_{i_1}, i_2, M_{i_2}, \ldots, i_T, M_{i_T}\}$$

where $R_n$ is the sequence number of this data report, $L_E$ is the event location, $t$ is the event time, $E$ is the event content, $i_j$ is the index of the key, and $M_{i_j}$ is the $MAC$ which is computed by corresponding key.

To facilitate the authenticating of false data dropping, before a data report is sent out or forwarded by a node, the data report will be copied and stored for a predefined period $T_c$.

**En-route filtering phase.** Once a node received a data report, the SEF scheme is used to filter false data reports as described in Figure 2, once a false data dropping is reported by a forwarding node. The en-route trap scheme together with the trust model and CND
scheme will work together to detect compromised nodes as described in former sections.

**Sink verification.** Once the sink receives a report, it can verify the correctness of every MAC since it has all the keys. Any forged MAC that eludes the en-route filtering by chance will be detected and dropped. The sink serves as the final defense that catches false data reports not filtered out by en-route nodes.

**Energy consumption and storage overhead**

**Energy consumption.** Compared with SEF, the extra energy consumption of ADET comes from three sources as listed below:

1. The first is the communication overhead of key indexes exchange between neighbors in the bootstrapping phase.
2. The second is the communication overhead between neighbors and computation overhead of trust model in the en-route trap.
3. The third is the communication overhead between neighbors in the CND algorithm.

As the research study\(^{21}\) pointed out, the energy consumption of MAC computing is much smaller than that of reports transmitting, similarly, the energy consumption of trust evaluation is also negligible. Moreover, the energy consumption of transmission of small size packets between neighbors is also much smaller than that of reports transmitting.

The key indexes exchange between neighbors could be incorporated in the neighbor discover process of SEF; thus, the first source of extra energy consumption is negligible. The communication overhead between neighbors in the en-route trap and CND algorithm is negligible compared with data reports forwarding; thus, the energy consumption of second source and third source is tolerable.

In fact, the main source of energy consumption in sensor network comes from the data reports transmitting and receiving: our ADET scheme requires no additional field in the data report compared with SEF. In summary, the extra necessary energy consumption involved in our ADET is affordable, since our ADET could detect and isolate compromised nodes effectively. The simulation results verified this.

**Storage overhead.** Compared with SEF, the extra storage overhead of each node in ADET comes from two sources as listed below:

1. The first is the storage overhead to store each neighbor’s \(k\) key indexes.
2. The second is the storage overhead to store each neighbor’s trust evaluation, number of negative behaviors, and number of positive behaviors in the trust model.

In SEF, each node needs to store \(k\) keys. In ADET, the first extra storage overhead of each node is to store each neighbor’s \(k\) key indexes. The second extra storage overhead of each node is to store each neighbor’s trust evaluation, number of negative behaviors, and number of positive behaviors in the trust model. Let \(N\) denotes the total number of nodes in sensor network, \(r\) denotes the communication range of sensor node, and \(R\) denotes the radius of sensor network. Then, the mathematical expectation of number of neighbors of each node can be denoted as

\[
E(N_n) = N \frac{r^2}{R^2}
\]  

(8)

Accordingly, the mathematical expectation of extra storage of each node can be denoted as

\[
E(S_{\text{extra}}) = (k \times L_1 + L_2 + 2L_3)N \frac{r^2}{R^2}
\]  

(9)

where \(L_1\) denotes the size of a key index, \(k\) denotes the number of keys owned by each node, \(L_2\) denotes the size of a trust evaluation, and \(L_3\) denotes the size of number of negative behaviors or number of positive behaviors.

In the simulation experiment, we set \(N = 1200\), \(k = 5\), \(r = 1\), \(R = 10\), \(L_1 = 1\) byte, \(L_2 = 4\) bytes, and \(L_3 = 2\) bytes, then the mathematical expectation of extra storage of each node \(E(S_{\text{extra}}) = 156\) bytes.

In summary, in ADET, each node incurs about 0.156 KB extra storage overhead compared with SEF, considering that ADET can detect compromise nodes, such extra storage overhead is tolerable. Current mainstream sensor nodes can meet the requirements of ADET (e.g. the MICA2 platform is equipped with 4 KB SRAM and 128 KB ROM).

**Simulation experiment and discussion**

The main goal of our ADET is to detect and isolate compromised nodes with low false positive rate. Another goal is that we prefer a lightweight scheme to detect the compromised nodes, which means our ADET does not introduce too much extra energy consumption and storage overhead. To test and develop our design, we could have used one of the several powerful simulators such as ns-2 or Opnet; all of them are well-known for having appropriate libraries for wireless networks. However, we developed a custom-made simulator in Java with a simplified network model because a controlled design of the network allows us to observe and analyze the effects of the design choices.
isolated from the interactions of physical, multi-access, and routing protocols.

Metrics
To evaluate the performance of ADET, we use the following metrics.

Detection power of en-route trap. The detection power of en-route trap is defined as the ratio of number of detected FFD attacks to total number of FFD attacks; note that a successful detection of FFD attack is defined as the compromised node which launch this attack is recorded a negative behavior by any normal nodes, and it is the main metric to evaluate the effectiveness of our en-route trap scheme. The detection power is supposed to vary between the upper and lower limits of theoretical value as described in former analysis.

Packet accuracy. This metric is defined by the formula as follows

\[ PA_n = 1 - \frac{NF_n}{NT_n} \]  

where \( PA_n \) denotes the packet accuracy at round \( n \), \( NF_n \) denotes the number of false data packets received by sink at round \( n \), and \( NT_n \) denotes the total number of data packets received by sink at round \( n \). We compared ADET with SEF by this metric, and the packet accuracy of our ADET is supposed to increase gradually with the detection and isolation of compromised nodes, while the packet accuracy of SEF is supposed to remain at a low level since it cannot detect compromised nodes.

Average residual energy level of normal nodes. The average residual energy level of normal nodes is the main metric to evaluate the energy efficiency of our ADET. We compare our ADET with SEF by this metric. As we analyzed in the energy consumption section, this metric of our ADET is supposed to remain at a little lower level compared to SEF.

Detection rate and false positive rate. The detection rate and false positive rate of compromised nodes are main metrics to evaluate the effectiveness of our ADET in detecting compromised nodes. The detection rate is supposed to increase gradually and the false positive rate is supposed to remain at a low level.

Average routing price of nodes. As we mentioned in the network model, the GEAR routing algorithm is adopted in our experiment, routing price is the main consideration to pick up next relay node. The average routing price of undetected compromised nodes is used to testify the existence of shutter islands as we mentioned in former section. It also figures out the reason of why some compromised nodes remain undetected.

Settings of parameters

Settings of \( T, n, m, and k \). \( T \) is the number of different key partitions required by a qualified data report, \( n \) is the number of key partitions of global key pool, \( m \) is the number of keys in each partition, and \( k \) is the number of keys assigned to a node before network deployment. All these parameters are empirical values, and we set \( T = 3, n = 10, m = 10, and k = 5 \) in the simulation experiment.

Settings of \( \theta, \omega, \) and \( \alpha \). \( \theta \) is the threshold of trust evaluation to trigger a CND process. \( \omega \) is the interaction weight of behavior which set to 1 in case of positive behavior and set to 0.5 in case of negative behavior in our trust model. \( \alpha \) is the probability to launch attack as described in the threat model. All of them are empirical parameters, and we set \( \theta = 0.5 \) and \( \alpha = 0.5 \) in the simulation experiment.

Simulation settings

In our simulation, 1200 nodes are distributed randomly in a circle area with a radius of 10 m, while sink locates at center of circle as described in Figure 15, each node has the same sensing range and communication range of 1 m, each node is assigned with a unique \( ID \) and five keys from one random key partition of the global key pool. The key pool is consisting of 10 key partitions, each key partition is consisting of 10 keys, and each key is consisting of a key index and a key value. The initial trust evaluation assigned to each neighbor node is 1, and the initial number of positive or negative behaviors is set to 1. Totally, 240 nodes are randomly assigned to be compromised nodes; it means 20% of the total nodes are compromised nodes. At the event area, the compromised node has the probability of \( \alpha \) to inject false data, or afford forged MAC to CoS. As a relay node, the compromised node has the probability of \( \alpha \) to launch FFD attack. The initial energy level of each node is set to 1, and each data report transmitting or receiving consumes 0.00005 in regard to energy level; note that data report transmitting and receiving is the main source of energy consumption according to related research works.21

The simulation experiment is carried out in 50 rounds; 12,000 events are simulated at random location within the network range at each round. All data reports will be forwarded to sink as described in our
ADET scheme, and with the joint work of en-route trap, trust model, and detection algorithm of compromised nodes, compromised nodes are detected and isolated. It is worth to mention that our ADET scheme is independent of routing algorithm, while the GEAR routing algorithm is used in the simulation experiment.

**Simulation results**

*Detection power of en-route trap.* It is shown in Figure 10 that the detection power of en-route trap in detecting FFD attack varies between the theoretical upper limit and lower limit as analyzed in section “Performance evaluation of en-route trap,” it testified to the effectiveness of our en-route trap in detecting FFD attack.

*Packet accuracy.* This section of simulation experiment compares ADET with SEF by packet accuracy. It is showed that at the beginning of simulation, both schemes have the same packet accuracy, with the experiments carried out round by round, the packet accuracy of ADET increases rapidly and reaches to 86%, while the packet accuracy of SEF remains at the same low level since it cannot detect and isolate compromised nodes. However, not all compromised nodes are detected in the experiment since the existence of shutter islands, it also explains the packet accuracy cannot rise to 100%. We leave the detection of compromised nodes in shutter islands to future works (Figure 11).

*Average residual energy level of normal nodes.* This section of simulation experiment compares two schemes by average residual energy level of normal nodes, and it is showed that our ADET scheme consumes a little more extra energy compared to SEF. The additional energy consumption came from the communications between neighbor nodes in the en-route trap scheme and detection of compromised nodes (Figure 12).

*Detection rate and false positive rate.* Figure 13 represents the detection rate and false positive rate of compromised nodes in ADET. The detection rate of compromised nodes increases rapidly and remains at 55%, it testified to the effectiveness of our ADET in detecting compromised nodes. However, the rest 45% of total compromised nodes remain undetected. This result also proved the existence of shutter islands which protected rest undetected compromised nodes in some extent as we explained in section “Discussion.” The false positive rate of our ADET remains at 0% which means no normal nodes are falsely detected as compromised nodes.

*Average routing price of nodes.* It is showed in Figure 14 that the average routing price of undetected compromised nodes increases rapidly and reaches to 7.3, while the average routing price of normal nodes remains at 6.8. This result also partially proved the existence of shutter islands.
Discussion

In this section, we focus on discussing the phenomenon of almost half of the total compromised nodes remain undetected in our ADET.

With the joint work of our en-route trap scheme and trust model, more and more compromised nodes are detected and isolated as showed in Figure 13. In some local areas, the relative shorter routes to sink are cut off due to the isolation of compromised nodes; thus, latter data reports must search another route to bypass these areas. As showed in Figure 14, the routing price of undetected compromised nodes within these areas increases rapidly; it also means compromised nodes within these areas have less opportunity to participate forwarding data reports. These areas together with areas at edge of sensor network form shutter islands; these shutter islands protected rest undetected compromised nodes from being detected in some extent. However, the rest undetected compromised nodes have less opportunity to attack sensor network since they locate at the edge of network.

The distribution of nodes at the beginning of experiment as showed in Figure 15 and at the end of experiment as showed in Figure 16 also testified to the existence of shutter islands. As we can see in Figure 16, most compromised nodes locate around center of network are detected and isolated, while most undetected compromised nodes locate at the network edge or shutter islands.

Conclusion

In this article, an ADET is proposed for detecting and isolating compromised nodes in a WSN. An active en-route trap scheme is proposed to trap compromised nodes in case of a false data dropping, a trust model is used to evaluate trust level of neighbor nodes with respect to their authentication behaviors, the detection power of en-route trap is analyzed, and the results of simulation experiment testified to the effectiveness of our ADET in detecting compromised nodes with a low false positive rate. Future prospects of this research are listed as follows:

1. The detection of the undetected compromised nodes locating at the network edge or shutter islands.
2. Design more schemes to trap compromised nodes beside the FFD attack, such as normal nodes cooperate to set a trap for compromised nodes.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported by the Excellent Youth Project of the Educational Department of Hunan Province of China under grant no. 18B410, the Natural Science...
ORCID iD
Jiang-Tao Wang https://orcid.org/0000-0001-8068-1093

References
1. Akyildiz IF, Su W, Sankarasubramaniam Y, et al. A survey on sensor networks. IEEE Commun Mag 2002; 40(8): 102–114.
2. Singla A and Sachdeva R. Review on security issues and attacks in wireless sensor networks. Int J Adv Res Comput Sci Softw Eng 2013; 3(4): 529–534.
3. Ye F, Luo H, Lu S, et al. Statistical en-route filtering of injected false data in sensor networks. IEEE J Select Area Commun 2005; 23: 839–850.
4. Yu L and Li J. Grouping-based resilient statistical en-route filtering for sensor networks. In: Proceedings of the IEEE INFOCOM, Rio de Janeiro, Brazil, 19–25 April 2009, pp.1782–1790. New York: IEEE.
5. Zhu S, Setia S, Jajodia S, et al. An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In: Proceedings of the ACM MobiHoc’05, 2005, https://www2.seas.gwu.edu/~cheng/388/LecNotes2007/LBRS.pdf
6. Zhu S, Setia S, Jajodia S, et al. An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In: Proceedings of the IEEE symposium on security and privacy, Berkeley, CA, 12 May 2004, pp.259–271. New York: IEEE.
7. Ye F, Yang H and Liu Z. Catching “moles” in sensor networks. In: Proceedings of the IEEE ICDCS’07, Toronto, ON, 25–27 June 2007. New York: IEEE.
8. Zhang Y, Yang J, Li W, et al. An authentication scheme for locating compromised sensor nodes in WSNs. J Netw Comput Appl 2010; 33(1): 50–62.
9. Xu X, Gao Z and Han L. An efficient compromised nodes detection system in wireless sensor networks. Int J Netw Secur 2018; 20(5): 960–970.
10. Seshadri A, Luk M, Shi E, et al. Pioneer: verifying code integrity and enforcing untampered code execution on legacy systems. In: Proceedings of the SOSP’05, Brighton, 23–26 October 2005. New York: ACM.
11. Spinellis D. Reflection as a mechanism for software integrity verification. ACM Trans Inform Syst Secur 2000; 3(1): 353383.
12. Yang Y, Wang X, Zhu S, et al. Distributed software-based attestation for node compromise detection in sensor networks. In: Proceedings of the IEEE SRDS, Beijing, China, 10–12 October 2007. New York: IEEE.
13. Lin X. CAT: building couples to early detect node compromise attack in wireless sensor networks. In: Proceedings of the IEEE GLOBECOM, Honolulu, HI, 30 November 4 December 2009. New York: IEEE.
14. Al-Riyami A, Zhang N and Keane J. An adaptive early node compromise detection scheme for hierarchical WSNs. IEEE Access 2016; 4: 4183–4206.
15. Iqbal Z, Nooshabadi S and Lee HN. Locating and disregarding the information from compromised sensors in a WSN. In: Proceedings of the 9th IEEE UEMCON, New York, 8–10 November 2018. New York: IEEE.
16. Neggazi B, Haddad M, Turau V, et al. A self-stabilizing algorithm for edge monitoring problem. In: Felber P and Garg V (eds) Stabilization, safety, and security of distributed systems. Cham: Springer, 2014, pp.93–105.
17. Remesh Babu Raman KR. Hybrid double layered security against data corruption and node compromise attacks in WSN. In: Proceedings of the IEEE ICJET, March 2014, pp.725–729, https://www.researchgate.net/publication/266393212_Hybrid_Double_Layered_Security_against_Data_Corruption_and_Node_Compromise_Attacks_in_WSN
18. Zhang Q, Yu T and Ning P. A framework for identifying compromised nodes in wireless sensor networks. ACM Trans Inform Syst Secur 2008; 11(3): 12.
19. Ho JW, Wright M and Das SK. ZoneTrust: fast zone-based node compromise detection and revocation in wireless sensor networks using sequential hypothesis testing. IEEE Trans Depend Secure Comput 2012; 9(4): 494–511.
20. Almenarez F, Marin A, Campo C, et al. PTM: a pervasive trust management model for dynamic open environments. In: Proceedings of the first workshop on pervasive security, privacy and trust (PSPT’04), 2004, https://www.researchgate.net/publication/228772564_PTM_A_pervasive_trust_management_model_for_dynamic_open_environments
21. Mao G, Fidan B and Anderson BDO. Wireless sensor network localization techniques. Comput Netw 2007; 51: 2529–2553.