In Wireless Sensor Networks Catching Packet Droppers and Modifiers

#V.Redya Jadav¹, Associate Professor, Computer Science and Engineering Department
#A.Mothilal², Assistant Professor, Computer Science and Engineering Department
#N.Srikanth³ M.Tech, Computer Science and Engineering Department
# Bomma Institute of Technology and Science, Khammam, A.P State, INDIA

Abstract:-

Though destructive to network functions, insider attackers are not detectable with only the classic cryptography based techniques. Many mission-critic sensor network applications demand an effective, light, flexible algorithm for internal adversary identification with only localized information available. The insider attacker detection scheme proposed in this paper meets all the requirements by exploring the spatial correlation existent among the networking behaviors of sensors in close proximity. Our work is exploratory in that the proposed algorithm considers multiple attributes simultaneously in node behavior evaluation, with no requirement on a prior knowledge about normal/malicious sensor activities. Moreover, it is application friendly, which employs original measurements from sensors and can be employed to monitor many aspects of sensor networking behaviors. Our algorithm is purely localized, fitting well to the large-scale sensor networks. Simulation results indicate that internal adversaries can be identified with a high accuracy and a low false alarm rate when as many as 25% sensors are misbehaving.

Packet dropping and modification are common attacks that can be launched by an adversary to disrupt communication in Wireless multi hop sensor networks. Many schemes have been proposed to mitigate or tolerate such attacks, but very few can effectively and efficiently identify the intruders. To address this problem, we propose a simple yet effective scheme, which can identify misbehaving forwarders that drop or modify packets. Extensive analysis and simulations have been conducted to verify the effectiveness and efficiency of the scheme.

So far sensor network broadcast protocols assume a trustworthy environment. However in safety and mission-critical sensor networks this assumption may not be valid and some sensor nodes might be adversarial. In these environments, malicious sensor nodes can deprive other nodes from receiving a broadcast message. We call this attack a denial-of-message attack (DoM). In this paper we model and analyze this attack, and present countermeasures. We present SIS, a secure implicit sampling scheme that permits a broadcasting base station to probabilistically detect the failure of nodes to receive its broadcast, even if these failures result from an attacker motivated to induce these failures undetectably. SIS works by eliciting authenticated acknowledgments from a subset of nodes per broadcast, where the subset is unpredictable to the attacker and tunable so as to mitigate acknowledgment implosion on the base station. We use a game-theoretic approach to evaluate this scheme in the face of an optimal attacker that attempts to maximize the number of nodes it denies the broadcast while remaining undetected by the base station, and show that SIS significantly constrains such an attacker even in sensor networks exhibiting high intrinsic loss rates. We also discuss...
extensions that permit more targeted detection capabilities.

Index Terms—Packet dropping, packet modification, intrusion detection, wireless sensor networks.

1. Introduction

In a wireless sensor network, sensor nodes monitor the environment, detect events of interest, produce data, and collaborate in forwarding the data toward a sink, which could be a gateway, base station, storage node, or querying user. Because of the ease of deployment, the low cost of sensor nodes and the capability of self-organization, a sensor network is often deployed in an unattended and hostile environment to perform the monitoring and data collection tasks. When it is deployed in such an environment, it lacks physical protection and is subject to node compromise. After compromising one or multiple sensor nodes, an adversary may launch various attacks [1] to disrupt the in-network communication. Among these attacks, two common ones are dropping packets and modifying packets, i.e., compromised nodes drop or modify the packets that they are supposed to forward.

To deal with packet droppers, a widely adopted countermeasure is multipath forwarding in which each packet is forwarded along multiple redundant paths and hence packet dropping in some but not all of these paths can be tolerated. To deal with packet modifiers, most of existing countermeasures aim to filter modified messages en-route within a certain number of hops. These countermeasures can tolerate or mitigate the packet dropping and modification attacks, but the intruders are still there and can continue attacking the network without being caught.

To locate and identify packet droppers and modifiers, it has been proposed that nodes continuously monitor the forwarding behaviors of their neighbors to determine if their neighbors are misbehaving, and the approach can be extended by using the reputation based mechanisms to allow nodes to infer whether a non neighbor node is trustable. This methodology may be subject to high-energy cost incurred by the promiscuous operating mode of wireless interface; moreover, the reputation mechanisms have to be exercised with cautions to avoid or mitigate bad mouth attacks and others. Recently, Ye et al. proposed a probabilistic nested marking (PNM) scheme [20]. But with the PNM scheme, modified packets should not be filtered out en route because they should be used as evidence to infer packet modifiers; hence, it cannot be used together with existing packet filtering schemes.

2. Related Work

We summarize the most related works along three major lines: detection of faulty sensor readings, detection of routing misbehavior, and detection of intrusion in wireless networks. Detection of event region or faulty sensors is explored for 0/1 decision predicate computation in [6][12][21]. The motivation comes from the observation that a remarkable change in sensor readings usually indicates a faulty sensor or a real event. The related algorithms require only the most recent readings (within a sliding window) of individual sensors. No collaboration among neighboring sensors is exploited. In [6], the “change point” of the time series is statistically computed. The result is used to answer questions such as “when does the front line of the contamination reach a location?” The detector proposed in [12] computes a running average and compares it with a threshold, which can be adjusted by a false alarm rate. In [21], kernel density estimators are designed to check whether the number of “abnormal” readings is beyond an application-specific threshold. The research on faulty sensor identification has been improved significantly in [9][27] by allowing any kind of scalar values as inputs instead of only 0/1 decision predicates. The detection algorithms in [9][27] can also infer faulty sensors from event sensors and compute the boundary of the event region. Similarly, our misbehaving sensor detection algorithm accepts any inputs expressed by real numbers. Further, our algorithm advances one more step in
identifying misbehaving sensors by considering multiple attributes simultaneously.

we propose a simple yet effective scheme to catch both packet droppers and modifiers. In this scheme, a routing tree rooted at the sink is first established. When sensor data are transmitted along the tree structure toward the sink, each packet sender or forwarder adds a small number of extra bits, which is called packet marks, to the packet. The format of the small packet marks is deliberately designed such that the sink can obtain very useful information from the marks. Specifically, based on the packet marks, the sink can figure out the dropping ratio associated with every sensor node, and then runs our proposed node categorization algorithm to identify nodes that are droppers/modifiers for sure or are suspicious droppers/modifiers. As the tree structure dynamically changes every time interval, behaviors of sensor nodes can be observed in a large variety of scenarios. As the information of node behaviors has been accumulated, the sink periodically runs our proposed heuristic ranking algorithms to identify most likely bad nodes from suspiciously bad nodes. This way, most of the bad nodes can be gradually identified with small false positive.

Our proposed scheme has the following features: 1) being effective in identifying both packet droppers and modifiers, 2) low communication and energy overheads, and 3) being compatible with existing false packet filtering schemes; that is, it can be deployed together with the false packet filtering schemes, and therefore it cannot only identify intruders but also filter modified packets immediately after the modification is detected. Extensive simulation on ns-2 simulator has been conducted to verify the effectiveness and efficiency of the proposed scheme in various scenarios.

3. SYSTEM MODEL

3.1 Network Assumptions

We consider a typical deployment of sensor networks, where a large number of sensor nodes are randomly deployed in a two dimensional area. Each sensor node generates sensory data periodically and all these nodes collaborate to forward packets containing the data toward a sink. The sink is located within the network. We assume all sensor nodes and the sink are loosely time synchronized [21], which is required by many applications. Attack resilient time synchronization schemes, which have been widely investigated in wireless sensor networks [22], [23], can be employed. The sink is aware of the network topology, which can be achieved by requiring nodes to report their neighboring nodes right after deployment.

3.2 Security Assumptions and Attack Model

We assume the network sink is trustworthy and free of compromise, and the adversary cannot successfully compromise regular sensor nodes during the short topology establishment phase after the network is deployed. This assumption has been widely made in existing work [8], [24]. After then, the regular sensor nodes can be compromised. Compromised nodes may or may not collude with each other. A compromised node can launch the following two attacks:

- Packet dropping. A compromised node drops all or some of the packets that is supposed to forward. It may also drop the data generated by itself for some malicious purpose such as framing innocent nodes.
- Packet modification. A compromised node modifies all or some of the packets that is supposed to forward. It may also modify the data it generates to protect itself from being identified or to accuse other nodes.
4. NETWORK MODEL AND ASSUMPTIONS

We consider a homogeneous sensor network with N sensors uniformly distributed in the network area. The network region is a b × b squared field located in the two dimensional Euclidean plane. All sensors have the same capabilities, and communicate through bidirectional links. We assume sensors in the proximity are burdened with similar workloads, thus nearby sensors are expected to behave similarly under normal conditions. An insider attacker is a sensor under the control of an adversary. It has the same network resource as a normal sensor, but its behavior is different compared to others. For example, an insider attacker may drop or broadcast excessive packets, report false readings that deviate significantly from other readings of neighboring sensors, etc. Throughout this paper, insider attackers are also called outliers or outlying sensors, while sensors working properly are called normal sensors. We assume each sensor works in promiscuous mode intermittently and listens on the channel for activities of direct neighbors. That is to say, sensor x can overhear the message to and from the immediate neighbor xi no matter whether or not x is involved in the communication. The monitoring is conducted intermittently, and xi’s networking behavior is modeled by a q-component attribute vector f(xi) = (f1(xi), f2(xi), ..., fq(xi))T with each component describing xi’s activity in one aspect. For each fixed j (1 ≤ j ≤ q), the component fj(xi) represents the actual monitoring result, such as the number of packets being dropped or broadcasted in one unit time, the actual reading of temperature/light/sound, the number of occurrences of some phenomenon, and so on. Therefore, fj(xi) can be continuous or discrete. For convenience, we assume that in any local area of the sensor field, all f(xi), where xi’s are normal sensors, follow the same multivariate normal distribution. (See details on this assumption in Subsection IV-C.)

After an internal adversary is detected, a report should be generated to the base station. Each sensor should exclude the outlying sensors in selecting the next-hop forwarder to realize the secure routing. In this paper we focus on the detection of insider attackers, thus report generation/delivery to the base station and outliers isolation will not be considered. In addition, we assume there exists a MAC layer protocol to coordinate neighboring broadcastings such that no collision occurs.

5. THE PROPOSED SCHEME

Our proposed scheme consists of a system initialization phase and several equal-duration rounds of intruder identification phases.

- In the initialization phase, sensor nodes form a topology which is a directed acyclic graph (DAG). A routing tree is extracted from the DAG. Data reports follow the routing tree structure.

- In each round, data are transferred through the routing tree to the sink. Each packet sender/forwarder adds a small number of extra bits to the packet and also encrypts the packet. When one round finishes, based on the extra bits carried in the received packets, the sink runs a node categorization algorithm to identify nodes that must be bad (i.e., packet droppers or modifiers) and nodes that are suspiciously bad (i.e., suspected to be packet droppers and modifiers).

- The routing tree is reshaped every round. As a certain number of rounds have passed, the sink will have collected information about node behaviors in different routing topologies. The information includes which nodes are bad for sure, which nodes are suspiciously bad, and the nodes’ topological relationship. To further identify bad nodes from the potentially large number of suspiciously bad nodes, the sink runs heuristic ranking algorithms.
In the following sections, we first present the algorithm for DAG establishment and packet transmission, which is followed by our proposed categorization algorithm, tree structure reshaping algorithm, and heuristic ranking algorithms. To ease the presentation, we first concentrate on packet droppers and assume no node collusion. After that, we present how to extend the presented scheme to handle node collusion and detect packet modifiers, respectively.

5.1 DAG Establishment and Packet Transmission

All sensor nodes form a DAG and extract a routing tree from the DAG. The sink knows the DAG and the routing tree, and shares a unique key with each node. When a node wants to send out a packet, it attaches to the packet a sequence number, encrypts the packet only with the key shared with the sink, and then forwards the packet to its parent on the routing tree. When an innocent intermediate node receives a packet, it attaches a few bits to the packet to mark the forwarding path of the packet, encrypts the packet, and then forwards the packet to its parent. On the contrary, a misbehaving intermediate node may drop a packet it receives. On receiving a packet, the sink decrypts it, and thus finds out the original sender and the packet sequence number. The sink tracks the sequence numbers of received packets for every node, and for every certain time interval, which we call a round, it calculates the packet dropping ratio for every node. Based on the dropping ratio and the knowledge of the topology, the sink identifies packet droppers based on rules we derive. In detail, the scheme includes the following components, which are elaborated in the following.

5.1.1 System Initialization

The purpose of system initialization is to set up secret pair wise keys between the sink and every regular sensor node, and to establish the DAG and the routing tree to facilitate packet forwarding from every sensor node to the sink. Preloading keys and other system parameters. Each sensor node $u$ is preloaded the following information:

- $K_u$: a secret key exclusively shared between the node and the sink.
- $L_r$: the duration of a round.
- $N_p$: the maximum number of parent nodes that each node records during the DAG establishment procedure.
- $N_s$: the maximum packet sequence number. For each sensor node, its first packet has sequence number 0, the $N_s$th packet is numbered $N_s - 1$, the $(N_s + 1)$th packet is numbered 0, and so on and so forth.

Topology establishment.

After deployment, the sink broadcasts to its one-hop neighbors a 2-tuple $h_0; 0i$. In the 2-tuple, the first field is the ID of the sender (we assume the ID of sink is 0) and the second field is its distance in hop from the sender to the sink. Each of the remaining nodes, assuming its ID is $u$, acts as follows:

1. On receiving the first 2-tuple $h_v; d_v$, node $u$ sets its own distance to the sink as $d_u = d_v + 1$.
2. Node $u$ records each node $w$ (including node $v$) as its parent on the DAG if it has received $h_w; d_w$ where $d_w = d_v$. That is, node $u$ records as its parents on the DAG the nodes whose distance (in hops) to the sink is the same and the distance is one hop shorter than its own. If the number of such parents is greater than $N_p$, only $N_p$ parents are recorded while others are discarded. The actual number of parents it has recorded is denoted by $n_p; u$. 
3. After a certain time interval, node $u$ broadcasts 2-tuple $h_u$; $d_u$ to let its downstream one-hop neighbors to continue the process of DAG establishment. Then, among the recorded parents on the DAG, node $u$ randomly picks one (whose ID is denoted as $P_u$) as its parent on the routing tree. Node $u$ also picks a random number (which is denoted as $R_u$) between 0 and $N_p - 1$. As to be elaborated later, random number $R_u$ is used as a short ID of node $u$ to be attached to each packet node $u$ forwards, so that the sink can trace out the forwarding path.
Finally, node u sends Pu, Ru and all recorded parents on the DAG to the sink. After the above procedure completes, a DAG and a routing tree rooted at the sink is established. The routing tree is used by the nodes to forward sensory data until the tree changes later; when the tree needs to be changed, the new structure is still extracted from the DAG. The lifetime of the network is divided into rounds, and each round has a time length of Lr. After the sink has received the parent lists from all sensor nodes, it sends out a message to announce the start of the first round, and the message is forwarded hop by hop to all nodes in the network. Note that, each sensor node sends and forwards data via a routing tree which is implicitly agreed with the sink in each round, and the routing tree changes in each round via our tree reshaping algorithm.

5.1.2 Packet Sending and Forwarding

Each node maintains a counter Cp which keeps track of the number of packets that it has sent so far. When a sensor node u has a data item D to report, it composes and sends the following packet to its parent node Pu:

$$\langle P_u, \{R_u, u, C_p \mod N_s, D, padu_{0}, padu_{1}\}_K_u \rangle,$$

where Cp MOD Ns is the sequence number of the packet. Ru (0 < Ru < Np -1) is a random number picked by node u during the system initialization phase, and Ru is attached to the packet to enable the sink to find out the path along which the packet is forwarded. fXgY represents the result of encrypting X using key Y.

Paddings padu:0 and padu:1 are added to make all packets equal in length, such that forwarding nodes cannot tell packet sources based on packet length. Meanwhile, the sink can still decrypt the packet to find out the actual content. To satisfy these two objectives simultaneously, the paddings are constructed as follows:

- For a packet sent by a node which is h hops away from the sink, the length of padu:1 is log(Np)*(h-1) bits. As to be described later, when a packet is forwarded for one hop, log(Np) bits information will be added and meanwhile, log(Np) bits will be chopped off.

  - Let the maximum size of a packet be Lp bits, a node ID be Lid bits and data D be LD bits. padu:0 should be Lp - Lid * 2 - log(Np)*h -log(Ns) - LD bits , where Lid - 2 bits are for Pu and u fields in the packet, field Ru is log(Np) bits long, field padu:1 is log(Np)*(h-1) bits long, and Cp MOD Ns is log(Ns) bits long. Setting padu:0 to this value ensures that all packets in the network have the same length Lp. When a sensor node v receives packet hv;mi, it composes and forwards the following packet to its parent node Pv:

$$\langle P_v, \{R_v, m'\}_K_v \rangle,$$

where m’ is obtained by trimming the rightmost log(Np) bits off m. Meanwhile, Rv, which has logNp bits, is added to the front of m’. Hence, the size of the packet remains unchanged.

Suppose on a routing tree, node u is the parent of node v and v is a parent of node w. When u receives a packet from v, it cannot differentiate whether the packet is originally sent by v or w unless nodes u and v collude. Hence, the above packet sending and forwarding scheme results in the difficulty to launch selective dropping, which is leveraged in locating packet droppers. We take special consideration for the collusion scenarios, which are to be elaborated later.

5.1.3 Packet Receiving at the Sink

We use node 0 to denote the sink. When the sink receives a packet (0, m’), it conducts the following steps:

1. Initialization. Two temporary variables u and m are introduced. Let u =0 and m = m’ initially.
2. The sink attempts to find out a child of node u, denoted as v, such that dec(Kv,m) results in a string starting with Rv, where dec(Kv,m) means the result of decrypting m with key Kv.
3. If the attempt fails for all children nodes of node $u$, the packet is identified as having been modified and thus should be dropped.
4. If the attempt succeeds, it indicates that the packet was forwarded from node $v$ to node $u$.

Now, there are two cases:

- If $\text{dec}(K_v, m)$ starts with $(R_v, v)$ it indicates that node $v$ is the original sender of the packet. The sequence number of the packet is recorded for further calculation and the receipt procedure completes.
- Otherwise, it indicates that node $v$ is an intermediate forwarder of the packet. Then, $u$ is updated to be $v$, $m$ is updated to be the string obtained by trimming $R_v$ from the leftmost. Then, steps 2-4 are repeated.

The process of packet receipt at the sink can be formalized as Algorithm 1. An example is provided in Section 3.1.4 in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.17. In this example, we show how the packet is sent and forwarded as well as how the packet is processed by the sink.

Algorithm 1. Packet Receipt at the Sink

5.2 Node Categorization Algorithm

In every round, for each sensor node $u$, the sink keeps track of the number of packets sent from $u$, the sequence numbers of these packets, and the number of flips in the sequence numbers of these packets, (i.e., the sequence number changes from a large number such as $N_{u} - 1$ to a small number such as $0$). In the end of each round, the sink calculates the dropping ratio for each node $u$. Suppose $N_{u, max}$ is the most recently seen sequence number, $N_{u, flip}$ is the number of sequence number flips, and $N_{u, rcv}$ is the number of received packets. The dropping ratio in this round is calculated as follows:

$$d_{u} = \frac{n_{u,flip} * N_{s} + n_{u, max} + 1 - n_{u, rcv}}{n_{u,flip} * N_{s} + n_{u, max} + 1},$$

Based on the dropping ratio of every sensor node and the tree topology, the sink identifies the nodes that are droppers for sure and that are possibly droppers. For this purpose, a threshold $O$ is first introduced. We assume that if a node’s packets are not intentionally dropped by forwarding nodes, the dropping ratio of this
node should be lower than \( \_ \). Note that \( \_ \) should be greater than 0, taking into account droppings caused by incidental reasons such as collisions.

The first step of the identification is to mark each node with “+” if its dropping ratio is lower than 0, or with “-” otherwise. After that, for each path from a leaf node to the sink, the nodes’ mark pattern in this path can be decomposed into any combination of the following basic patterns, which are also illustrated by Fig. 1:

![Node status pattern](image)

**6. Tree Reshaping and Ranking Algorithms**

The tree used to forward data is dynamically changed from round to round, which enables the sink to observe the behavior of every sensor node in a large variety of routing topologies. For each of these scenarios, node categorization algorithm is applied to identify sensor nodes that are bad for sure or suspiciously bad. After multiple rounds, sink further identifies bad nodes from those that are suspiciously bad by applying several proposed heuristic methods.

**6.1 Tree Reshaping**

The tree used for forwarding data from sensor nodes to the sink is dynamically changed from round to round. In other words, each sensor node may have a different parent node from round to round. To let the sink and the nodes have a consistent view of their parent nodes, the tree is reshaped as follows. Suppose each sensor node \( u \) is preloaded with a hash function \( h() \) and a secret number \( K_u \) which is exclusively shared with the sink. At the beginning of each round \( i \) (\( i \leq 1, 2, \ldots \)), node \( u \) picks the \( \frac{1}{2}h_i(K_u) \ MOD n_p;u \) th parent node as its parent node for this round, where \( h_i(K_u) \) \( \frac{1}{2}h_0h_i_1K_u \) and \( n_p;u \) is the number of candidate parent nodes that node \( u \) recorded during the tree establishment phase. Recall that node \( u \)’s candidate parent nodes are those which are one hop closer to the sink and within node \( u \)’s communication range. Therefore, if node \( u \) choose node \( w \) as its parent in a round, node \( w \) will not select node \( u \) as its parent, and the routing loop will not occur. Note that, how the parents are selected is predetermined by both the preloaded secret \( K_u \) and the list of parents recorded in the tree establishment phase. The selection is implicitly agreed between each node and the sink. Therefore, a misbehaving node cannot arbitrarily select its parent in favor of its attacks.

**6.2 Identifying Most Likely Bad Nodes from Suspiciously Bad Nodes**

We rank the suspiciously bad nodes based on their probabilities of being bad, and identify part of them as most likely bad nodes. Specifically, after a round ends, the sink calculates the dropping ratio of each node, and runs the node categorization algorithm specified as Algorithm 2 to identify nodes that are bad for sure or suspiciously bad. Since the number of suspiciously bad nodes is potentially large, we propose how to identify most likely bad nodes from the suspiciously bad nodes as follows. By examining the rules in Cases 3 and 4 for identifying suspiciously bad nodes, we can observe that in each of these cases, there
are two nodes having the same probability to be bad and at least one of them must be bad. We call these two nodes as a suspicious pair. For each round $i$, all identified suspicious pairs are recorded in a suspicious set denoted as

$$S_i = \{\{u_j, v_j\} | \{u_j, v_j\} \text{ is a suspicious pair and } \{u_j, v_j\} = \{v_j, u_j\}\}.$$ 

Therefore, after $n$ rounds of detection, we can obtain a series of suspicious sets: $S_1; S_2; \ldots ; S_n$. We define $S$ as the set of most likely bad nodes identified from $S_1; S_2; \ldots ; S_n$, if $S$ has the following properties:

- Coverage. $\forall (u, v) \in S_i$ ($i = 1; \ldots ; n$), it must hold that either $u \in S$ or $v \in S$. That is, for any identified suspicious pair, at least one of the nodes in the pair must be in the set of most likely bad nodes. WANG ET AL.: CATCHING PACKET DROPPERS AND MODIFIERS IN WIRELESS SENSOR NETWORKS 839
- Most-likeliness. $\forall (u, v) \in S_i$ ($i = 1; \ldots ; n$), if $u \in S$ but $v \notin S$, then $u$ must have higher probability to be bad than $v$ based on $n$ rounds of observation.
- Minimality. The size of $S$ should be as small as possible in order to minimize the probability of miscuing innocent nodes.

Among the above three conditions, the first one and the third one can be relatively easily implemented and verified. For the second condition, we propose several heuristics to find nodes with most-likeliness.

Global ranking-based (GR) method. The GR method is based on the heuristic that, the more times a node is identified as suspiciously bad, the more likely it is a bad node. With this method, each suspicious node $u$ is associated with an accused account which keeps track of the times that the node has been identified as suspiciously bad nodes. To find out the most likely set of suspicious nodes after $n$ rounds of detection, as described in Algorithm 3, all suspicious nodes are ranked based on the descending order of the values of their accused accounts. The node with the highest value is chosen as a most likely bad node and all the pairs that contain this node are removed from $S_1; \ldots ; S_n$, resulting in new sets. The process continues on the new sets until all suspicious pairs have been removed. The GR method is formally presented in Algorithm.

Algorithm: The Global Ranking-Based Approach

1: Sort all suspicious nodes into queue $Q$ according to the descending order of their accused account values
2: $S \leftarrow \emptyset$
3: while $\bigcup_{i=1}^{n} S_i \neq \emptyset$ do
4: $u \leftarrow \text{dequeue}(Q)$
5: $\overline{S} \leftarrow \overline{S} \setminus \{u\}$
6: remove all $\langle u, \ast \rangle$ from $\bigcup_{i=1}^{n} S_i$

Stepwise ranking-based (SR) method. It can be anticipated that the GR method will falsely accuse innocent nodes that have frequently been parents or children of bad nodes: as parents or children of bad nodes, according to previously described rules in Cases 3 and 4, the innocents can often be classified as suspiciously bad nodes. To reduce false accusation, we propose the SR method. With the SR method, the node with the highest accused account value is still identified as a most likely bad node. However, once a bad node $u$ is identified, for any other node $v$ that has been suspected together with node $u$, the value of node $v$’s accused account is reduced by the times that $u$ and $v$ have been suspected together. This adjustment is motivated by the possibility that $v$ has been framed by node $u$. After the adjustment, the node that has the highest value of accused account among the rest nodes is identified as the next mostly likely bad node. This process continues until all suspicious pairs have been removed. The SR method is formally presented in Algorithm.
Algorithm: The Stepwise Ranking-Based Approach

1: \( \overline{S} \leftarrow \emptyset \)
2: while \( \bigcup_{i=1}^{n} S_i \neq \emptyset \) do
3: \( u \leftarrow \) the node has the maximum times accused in \( S_1, \ldots, S_n \)
4: \( \overline{S} \leftarrow \overline{S} \setminus \{u\} \)
5: remove all \( \langle u, \ast \rangle \) from \( \bigcup_{i=1}^{n} S_i \)

Hybrid ranking-based (HR) method. The GR method can detect most bad nodes with some false accusations while the SR method has fewer false accusations but may not detect as many bad nodes as the GR method. To strike a balance, we further propose the HR method, which is formally presented in Algorithm 5. According to HR, the node with the highest accused account value is still first chosen as most likely bad node. After a most likely bad node has been chosen, the one with the highest accused account value among the rest is chosen only if the node has not always been accused together with the bad nodes that have been identified already. Thus, the accusation account value is considered as an important criterion in identification, as in the GR method; meanwhile, the possibility that an innocent node being framed by bad nodes is also considered by not choosing the nodes which are always being suspected together with already identified bad nodes, as in the SR method. The HR method is formally presented in Algorithm.

Algorithm: The Hybrid Ranking-Based Approach

1: Sort all suspicious nodes into queue \( Q \) according to descending order of their accused account values
2: \( \overline{S} \leftarrow \emptyset \)
3: while \( \bigcup_{i=1}^{n} S_i \neq \emptyset \) do
4: \( u \leftarrow \text{deque}(Q) \)
5: if there exists \( \langle u, \ast \rangle \in \bigcup_{i=1}^{n} S_i \) then
6: \( \overline{S} \leftarrow \overline{S} \setminus \{u\} \)
7: remove all \( \langle u, \ast \rangle \) from \( \bigcup_{i=1}^{n} S_i \)

6.3 Handling Collusion

Because of the deliberate hop by hop packet padding and encryption, the packets are not distinguishable to the upstream compromised nodes as long as they have been forwarded by an innocent node. The capability of launching collusion attacks is thus limited by the scheme. However, compromised nodes that are located close with each other may collude to render the sink to accuse some innocent nodes. We discuss the possible collusion scenarios in this section and propose strategies to mitigate the effects of collusion.

7. PERFORMANCE EVALUATION

The effectiveness and efficiency of the proposed scheme are evaluated in the ns-2 simulator (version 2.30). The detailed performance metric, methodology as well as the attack models is discussed in section 4.1 in the supplementary file, available in the online supplemental material. The simulation results are presented in the supplementary file, available in the online supplemental material. We first study the impact of various system parameters on the detection. When there is no collusion. We then evaluate our proposed scheme under node collusion attacks. To identify packet modifiers and droppers, it has been proposed to add nestedMACsto address this problem in [20] and [25]. We compare our proposed scheme with the PNM scheme [20] regarding detection performance and communication overhead. Details are presented in the supplementary file, available in the online supplemental material.

As the proposed scheme outperforms the PNM scheme in terms of detection performance and communication overhead, we further measure the computational overhead of the packet sending and forwarding scheme on TelosB motes, which are widely used resource-constrained sensor motes [26]. Details are shown in Section 4.4 in the supplementary file, available in the online supplemental material.
8. CONCLUSIONS

We propose a simple yet effective scheme to identify misbehaving forwarders that drop or modify packets. Each packet is encrypted and padded so as to hide the source of the packet. The packet mark, a small number of extra bits, is added in each packet such that the sink can recover the source of the packet and then figure out the dropping ratio associated with every sensor node. The routing tree structure dynamically changes in each round so that behaviors of sensor nodes can be observed in a large variety of scenarios. Finally, most of the bad nodes can be identified by our heuristic ranking algorithms with small false positive. Extensive analysis, simulations, and implementation have been conducted and verified the effectiveness of the proposed scheme.

In this paper we propose a novel idea of insider attacker detection in wireless sensor networks. By exploiting the spatial correlation among the networking behaviors of sensors in close proximity, our detection algorithm can achieve a high detection accuracy and a low false alarm rate as indicated by the extensive simulation study. The nice feature of the algorithm is that it requires no prior knowledge about normal or malicious sensors, which is important considering the dynamic attacking behaviors. Further, our algorithm can be employed to inspect any aspects of networking activities, with the multiple attributes evaluated simultaneously. The algorithm is pure localized, thus scales well to large sensor networks. We notice that the detection algorithm can be specialized by exploring the degree of the correlations existent among different aspects of sensor networking behaviors. We target this specialization as a future work.

9. REFERENCES


V. Redya received the B.E. degrees in computer science and engineering from Osmania University, Hyderabad, AP, and India. He is received M.Tech in Computer Science and Engineering at Jawaharlal Nehru Technological University, Hyderabad, AP and India. He is pursuing Ph.D program in computer science and Engineering.
A.Mothilal received the B.Tech, degrees in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, AP, and India. He is received M.Tech in Computer Science and Engineering from Jawaharlal Nehru Technological University, Hyderabad, AP and India.

N.Srikanth received the B.Tech, degrees in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, AP, and India. He is Pursuing M.Tech in Computer Science and Engineering at BOMMA INSTITUTE OF TECHNOLOGY and SCIENCE from Jawaharlal Nehru Technological University, Hyderabad, AP and India.