ComSen: A Detection System for Identifying Compromised Nodes in Wireless Sensor Networks

Yi-Tao Wang
Department of Computer Science
University of California, Los Angeles
Email: yitao@cs.ucla.edu

Rajive Bagrodia
Department of Computer Science
University of California, Los Angeles
Email: rajive@cs.ucla.edu

***Abstract*—Wireless sensor networks (WSNs) are vulnerable to adversaries as they are frequently deployed in open and unattended environments. Adversaries can extract vital information, such as security keys, from compromised nodes and use them to launch insider attacks, so detecting when a node is compromised is important to securing WSNs. In this paper, we present ComSen, an accurate and lightweight intrusion detection system for identifying compromised nodes in wireless sensor networks. Through quantitative results, we demonstrate the benefits of ComSen: (1) it is not vulnerable to slander attacks, (2) it provides detection rates of 99% and false positive ratios of less than 2% in environments with loss rates of 30%, which cannot be achieved by existing systems, (3) it runs on most WSNs because it uses common application features (sensor readings, receive power, send rate, and receive rate) and can adjust its detection behavior if the sensor application doesn't have periodic transmissions or lacks inter-node communication, and (4) it has low memory, computation, and communication overheads that allows it to scale to networks of over thousands of nodes.**

***Keywords*-wireless sensor network; ComSen; intrusion detection system;**

I. INTRODUCTION

The inexpensive and autonomous nature of sensor nodes, called motes, have allowed wireless sensor networks (WSNs) to expand to many areas, including habitat monitoring, healthcare applications, home automation, and traffic control. However, as WSNs expand to more security-critical applications like battlefield surveillance, securing them against adversaries becomes an important concern. Without security in hostile environments, it is impossible to trust the reports from WSNs.

However, motes have limited resources, in terms of computational power, memory, and battery life, and are frequently deployed in open environments, so they are vulnerable to node compromise (where an adversary gains control of a node in the network). Without detection by the network, the compromised node is considered an authorized participant in the network and can launch insider attacks, which are attacks that leverage their higher access and authority.

Thus, security-critical WSNs must deploy measures to guard against node compromise. In general, security strategies can be classified as prevention, detection, or recovery. Since motes are designed to be small and cheap, their resource constraints limit the effectiveness of prevention mechanisms in WSNs [2], [15], [19]. Adversaries can physically compromise nodes using more powerful machines, such as laptops, so prevention will only delay attackers.

Detection provides a defense against compromises that bypass any deployed preventive measures. If the compromise is detected quickly, then appropriate measures can be taken to limit the damage of compromised nodes. Although there are several approaches that focus on detecting compromised nodes, they all suffer from various shortcomings. Most approaches are targeted at specific scenarios and perform poorly otherwise. For example, most systems do not account for lossy environments, common in WSNs [1], [19], and a 20% loss rate will decrease the detection rate by more than 50% and increase the false positive ratio to over 90% [28].

In this paper, we present ComSen, an intrusion detection system for identifying compromised nodes in WSNs that satisfies all of the following criteria:

- **Accuracy:** ComSen accurately detects compromised nodes in WSNs in a timely manner. Specifically, this requirement involves (1) a high detection rate, (2) a low false positive rate, and (3) a low detection time. A high detection rate means that most compromises are reported. Few false positives means that the majority of reported compromises are valid (i.e., they correspond to actual compromises) so actions against any reported compromised node can be taken with confidence. Lastly, a low detection time limits the malicious actions that can be performed by compromised nodes before they are detected.
- **Flexibility:** ComSen is not tailored for a specific application or deployment, which would limit its applicability. It makes few as assumptions about the underlying network as possible and can be used in the majority of deployments to detect compromises.
- **Robustness:** Compromised nodes may attempt to undermine the detection system through malicious behavior, such as slander attacks, where they send false information that implicates legitimate nodes as compromised. ComSen must guard against such malicious actions. Even with perfect knowledge of ComSen, an adversary will never be able to use ComSen to control the rest of the network.
- **Scalability:** Since motes have limited resources, applications with high overheads will interfere with other applications and decrease the lifespan of the mote [1]. ComSen provides low overhead so that its impact on any other applications deployed in the network is minimized.

Our primary goal in designing and implementing ComSen is

 ISBN: 978-1-61208-209-7

to improve the overall security of WSNs by providing a system for accurately identifying compromised nodes that has, thus far, been absent. ComSen provides a lightweight and distributed architecture that can be deployed on resource limited devices such as the nodes of a sensor network with minimal impact on other deployed applications and the lifespan of the network. It provides low overhead while maintaining its accuracy for WSNs by adapting its detection mechanism to leverage the characteristics of the underlying network. Through several experiments, we show that ComSen provides accurate detection of compromised nodes than other comparable systems and can scale up to thousands of nodes.

The rest of the paper is structured as follows: Section II goes over related work in detecting compromised nodes for WSNs. Section III discusses our system and threat models. ComSen's design is presented in Section IV, implementation in Section V, and evaluation in Section VI. Lastly, Section VII is the conclusion and future work.

II. Related Work

The majority of existing detection approaches for WSNs focus on specific attacks, such as replication [5], [21], wormhole [12], [14], and sybil [6], [13]. Although they may indirectly detect compromised nodes, adversaries can avoid detection by avoiding the target attack.

The traditional method of detecting compromised nodes is to use attestation. Attestation is the checking of all or random portions of a node's memory for changes that would exist if the node was running altered code. The advantage of this approach is that it is capable to detecting compromised nodes that are not misbehaving. Several software-based attestation techniques have been proposed for WSNs [22], [23], [24]. Unfortunately, secure software-based attestation has yet to be realized in WSNs and existing attestation techniques have been circumvented through various approaches, most of which have no hardware requirements beyond a laptop and serial cable [2].

Other approaches focus on monitoring communications for misbehavior. Misbehavior is decided using anomaly-based or rule-based approaches. Anomaly-based approaches establish a baseline behavior for neighbors and consider behavior that deviate from the baseline as anomalous. For example, Onat and Miri proposed an intrusion detection system (IDS) that monitors two features: (1) the packet arrival rate and (2) the receive power [20]. Nodes will continue to monitor these two features from neighbors and any new value that deviates a certain amount from the established baseline is considered anomalous. Malicious behavior that causes a change in either feature, such as a replay attack, would be detectable. Rule-based approaches detect misbehavior as soon as a condition, established before deployment of the network, is met. For example, COOL is an IDS that uses the relationship between incoming and outgoing messages to detect compromised nodes [30]. COOL's approach is based on the observation that the majority of outgoing messages should be forwards of incoming messages. Any node that is sending T more than it is receiving, where T is some threshold, is considered to be compromised.

Our work differs from existing approaches in that is offers improved flexibility, robustness, and scalability. ComSen provides accurate detection of compromised nodes in lossy environments and regardless of other applications deployed on the mote. It is secure against mass slander attacks, where compromised nodes collude to hinder the detection process. Furthermore, it has a low overhead that allows it to be deployed in WSNs of over a thousand nodes without affecting other deployed applications or significantly reducing the lifespan of the network.

III. System and Threat Model

We have designed ComSen based on the following common characteristics of WSNs and compromised nodes:

- The network has densely deployed sensor nodes such that sensor nodes have overlapping sensing ranges and a given event is detected by multiple nodes. Since the ranges overlap, a sensor node can monitor the activities of its neighbors.
- Sensor nodes are motes with limited computation, communication, and energy resources. For example, the Mica2 motes have a 4 MHz 8 bit Atmel microprocessor, and are equipped with an instruction memory of 128KB and a RAM of 4KB [17]. There are no mobile nodes because of mobility's high energy cost.
- There is a routing protocol used to forward messages between the base station and the nodes.
- The base station is a higher order device, such as a computer, that is in a secure location.
- Sensor nodes have unique identifiers so that the base station knows which reported compromise corresponds to which node.
- All messages have timestamps and nonces.
- Adversaries can compromise sensor nodes using physical capturing or through the radio communication channel. Once a sensor node is compromised, all information of the node, including any security keys, becomes available to the adversary.
- Although compromised nodes can perform any number of attacks to degrade the network's security and performance, we focus on compromised nodes that perform malicious acts (i.e., launch insider attacks such as falsifying data).

As we will show in Section IV, ComSen leverages several of the characteristics above to provide accurate identification of compromised nodes and low overhead.

IV. ComSen's Architecture

In designing ComSen, we had to choose whether to use a distributed, centralized, or hybrid approach. A purely distributed IDS is challenging because the resource constraints of sensor nodes prevent the use of complex algorithms that would provide more accurate results in a timely manner. For example, modular arithmetic, used in traditional security protocols like RSA, is expensive on the 8-bit processors of motes. Complex calculations could be distributed across multiple sensor nodes, but the node performing a critical calculation can be compromised, and the result falsified. In contrast, a centralized approach does not have these problems because the base station has more resources and

 ISBN: 978-1-61208-209-7

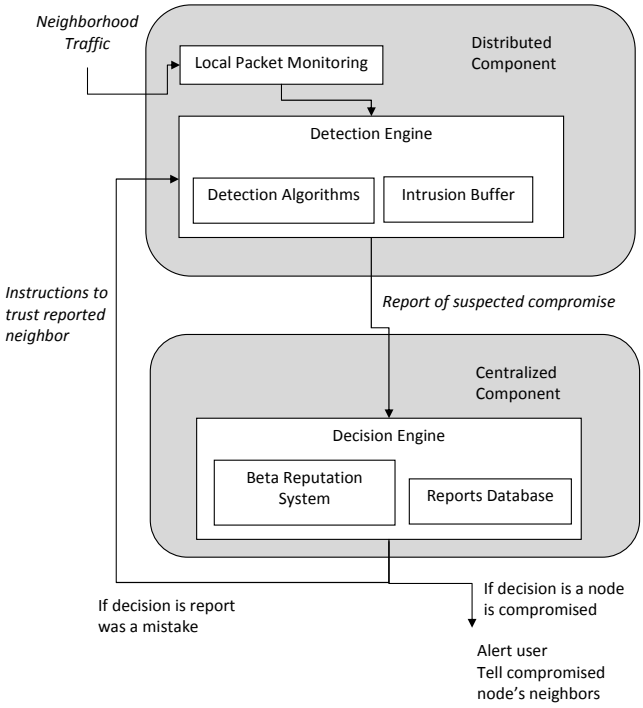


Fig. 1. Overview of ComSen

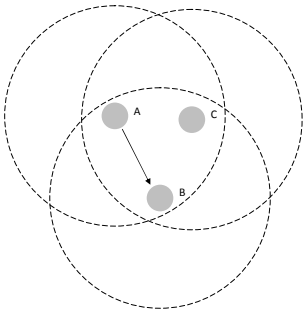


Fig. 2. Neighbor monitoring

is secure. However, the data received by the base station from compromised nodes can be false, so there must be some level of redundancy added to the network so that false data from a single node is detectable.

Thus, ComSen uses a hybrid approach, illustrated in Figure 1, that consists of two components: a distributed system running on every node in a WSN and a centralized system running on the base station.

- **Distributed Component:** A copy of this component runs on every sensor node, alongside the sensor applications, routing protocols, etc. Each copy is responsible for detecting *possible* compromises in neighboring nodes and reporting that to the base station. The detection relies on neighbor monitoring, where each node records and analyzes the behavior of its neighbors. This can be done without incurring any communication overhead because WSNs broadcast by nature. As Figure 2 illustrates, when one sensor node communicates with another, sent packets are overhead by neighboring nodes in radio range of the sender. Section IV-A discusses this component in detail.
- **Centralized Component:** The base station is a higher order machine, so ComSen uses it to perform complex analysis to decide if reports of possible compromises are accurate or mistakes. The base station aggregates data from the entire network to make a decision that would be impossible for sensor nodes given their local view and limited resources. The centralized component is discussed in Section IV-B.

There is an initial setup period after the network is deployed. During this period, nodes establish their neighbor lists, routes to the base station, etc. This period will not introduce any vulnerabilities as long as the network is secure for an amount of time after its initial deployment (i.e., adversaries are not able to immediately compromise a node in the network as soon as the network is deployed). However, if this requirement cannot be met, then the information must be pre-programmed onto each node before the network is deployed.

A. Distributed Component

Each sensor node, with a distributed component running on it, will record data from its neighbors and establish baselines based on the records. The data is from system features that reflect the behavior of nodes, so the baselines reflect the normal behavior of nodes. Every newly recorded behavior will be analyzed for *anomalies*, behavior that deviates from their established baselines; every instance of which is considered a misbehavior. If a neighbor continues to misbehave, it is considered compromised and a report about it is sent to the base station.

TO account for transient errors, such as collisions, and other non-malicious misbehavior, ComSen provides some flexibility in the amount of misbehavior that is tolerated before a neighbor is considered compromised. There is no collaboration between nodes to decide if a neighbor is misbehaving. This independent decision process means that compromised nodes cannot influence legitimate neighbors' views.

The system features that are monitored are discussed in Section IV-A1. Section IV-A2 discusses the algorithms that the distributed component uses to detect misbehavior.

1) Monitored Features: The initial step in designing any detection based security system is to select the system features that will be monitored. To provide support for most WSNs, ComSen only monitors some common features in WSNs:

- **Sensor Reading:** Nodes in WSNs are rarely deployed in scenarios where there is no correlation between the sensor readings of neighboring nodes (i.e., the readings of one node are not independent of that of its neighbors) [1]. By monitoring the sensor readings, we can detect attacks that attempt to distort gathered information.
- **Receive Power:** In static networks, the receive power should remain constant. Fluctuations may be caused by changes in the communication hardware or position of the corresponding node.
- **Send Rate:** Most applications take sensor readings and transmit them periodically. Any routed packets would also be periodic. Thus the rate of packets sent by nodes should follow a consistent pattern. Most attacks, such as selective forwarding, sybil, and replay, cause deviations in

 ISBN: 978-1-61208-209-7

this metric. Furthermore, sudden periods of inactivity may be caused by the process of adversaries re-programming nodes.

- **Receive Rate:** The ratio between incoming and outgoing packets should be constant because outgoing packets can only be packets being routed or packets generated by the node. A neighboring node that does not change its send rate when its receive rate changes may be compromised. It should be noted that, regardless of whether its data is encrypted, a packet's header (with the source and destination information) is often viewable by all nodes.

These choices allow ComSen to be widely applicable, because these features will be accessible in the majority of WSNs. However, these features may not be appropriate under two scenarios: (1) the packet data can only be decrypted by the base station and (2) the application rarely communicates information to the base station.

The first scenario arises when the confidentiality of information being transmitted in the WSN is important (i.e., not simple temperature readings). Since compromise cannot be detected and prevented in zero time, it's possible that some sent information may be eavesdropped by compromised nodes. Thus, packet data may be encrypted so that only the base station can decrypt it. Under such conditions, we can compensate for not monitoring the sensor readings by increasing the number of monitored neighbors to achieve comparable performance from ComSen at the cost of higher memory overhead. Section VI discusses this in further detail.

The second case is caused by applications with non-periodic communication. For example, a WSN in a demilitarization zone that only sends when movement is detected would have no communication under peaceful circumstances. Baselines cannot be established for most of the features because there's inadequate information being monitored. ComSen compensates by using an *activity mode* where nodes to send sensor readings at a rate that's a hash of its unique identification number. The communication overhead is necessary in order to prevent long periods of silence where nodes can be compromised and studied by adversaries without detection. Activity mode is only used when there is inadequate communication by the application.

2) Algorithms for Detecting Misbehavior: There are two categories of algorithms that detect misbehavior: anomaly-based and rule-based, both of which use records of monitored systems features. In anomaly-based algorithms, a baseline is established from the records and any new records that differ from the baseline, above a threshold, are considered anomalies. In contrast, rule-based check for a specific criteria (e.g., any two packets with the same header signals a replay attack). In ComSen, we decided to focus on anomaly-based algorithms in order to meet ComSen's flexibility criteria; rule-based algorithms target specific scenarios and the rules they use have to constantly updated for every new scenario.

The detection of misbehavior by the distributed component of ComSen can be divided into five algorithms: detection using (1) sensor reading, (2) receive power, (3) send rate, (4) receive rate, and (5) join messages. The first four are anomaly-based algorithms using the four features that are monitored (Section IV-A1). The last algorithm is rule-based.

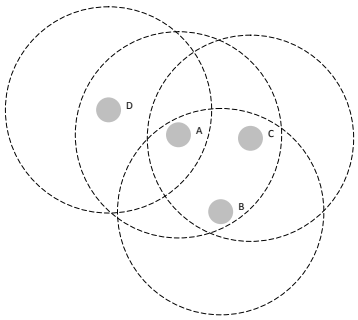


Fig. 3. Node A cannot impersonate any other node without one neighboring detecting a new neighbor

For the rule-based algorithm, if a node detects a new neighbor outside of the setup period, by hearing any messages from it. It will immediately consider the neighbor compromised.

These rules prevent compromised nodes, and even external attackers, from impersonating other nodes without being considered as new neighbors and reported. Figure 3 illustrates this property. Suppose node A was compromised and wants to impersonate another node. It cannot impersonate A or B without node D detecting a new neighbor. It cannot impersonate D without nodes B and C detecting a new neighbor. It cannot impersonate any other node except with B, C, and D detecting new neighbors. Thus, with enough neighbors monitoring each other, any attacks involving impersonation can be detected.

The four anomaly-based algorithms all follow a similar approach. Every node has two buffers for each monitored neighbor: a *packet buffer* and a *misbehavior buffer*. The buffers are shared by all four algorithms and use a sliding window approach where the last *N* packets/reports about the corresponding neighbor are stored. The data stored in the packet buffer is used to calculate that neighbor's baseline. New packets are compared against the baseline and any packets that deviate from the baseline by more than some threshold is considered anomalous. Anomalous packets indict misbehavior and cause the detecting node to generate a misbehavior report. All such reports are added to its misbehavior buffer. When the cumulative weight of reports in the misbehavior buffer passes a threshold, the node will report, to the base station, that the corresponding neighbor is compromised.

An overview of the algorithm that uses receive power is shown in Figure 4 and its corresponding equations are:

$$\begin{aligned} power_{new} - power_{max} > T, if\ power_{new} > power_{max} \\ power_{min} - power_{new} > T, if\ power_{new} < power_{min} \end{aligned} \quad (1)$$

The algorithm calculates the minimum and maximum values of packet receive power from the packet buffer. A new packet is anomalous if its receive power is T below the min or T above the max. Any detected anomalous packets are considered to be misbehaviors. Anomalous packets are added to the packet buffer so that anomalies caused by environmental changes can be accounted in future baseline calculations.

The algorithm that uses the sensor readings is almost identical to the one that uses the receive power. The only difference is

 ISBN: 978-1-61208-209-7

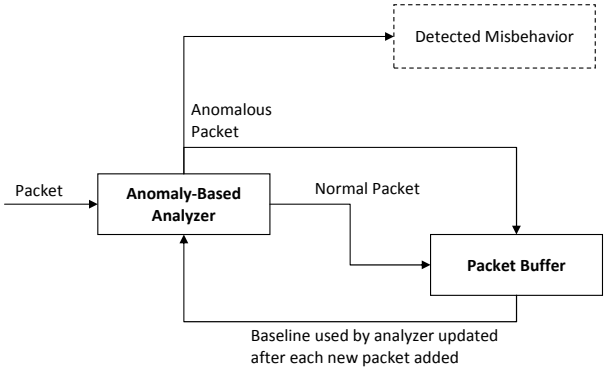


Fig. 4. Overview of the detection algorithms that use receive power and sensor readings

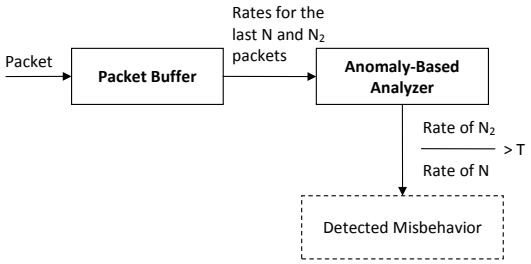


Fig. 5. Overview of the detection algorithms that use send rate and receive rate

that the difference between the node's and the neighbor's sensor readings is used instead of the receive power.

Figure 7 shows an overview of the algorithm that uses the send rate. It calculates two rates: the rate at which the last N_2 packets are sent (including the last packet), $rate_{N2}$, and the rate at which the last N packets are sent, $rate_N$ where $N > N_2$. If the ratio of these two rates is above a threshold K the corresponding neighbor is considered to be compromised.

The algorithm that uses the receive rate only differs in two ways. First, instead of counting packets sent by the neighbor, the counts are of packets received by the neighbor (i.e., sent to the neighbor). Second, the rates are replaced with send-receive ratios (i.e., $rate_{N2}$ becomes $rate_{sent_N2}/rate_{rec_N2}$ and $rate_N$ becomes $rate_{sent_N}\ /rate_{rec_N}$).

All reported misbehaviors generated by the four anomaly-based algorithms for a node are stored in a shared misbehavior buffer. Each reported misbehavior is given a weight based on the time that it was detected, t_{stamp}, and the current time, $t_{current}$. When a misbehavior for a neighbor is detected, the total weight of detected misbehaviors for that neighbor is calculated using:

$$\sum_{M}(t_{current} - t_{stamp}) + 0.3\sum_{m}(t_{current} - t_{stamp}) \qquad (2)$$

where M is all detected misbehaviors of the same type and m are all detected misbehaviors of all other types. When this total passes a threshold, T_M, the corresponding neighbor is considered compromised. Thousands of simulations, described in Section VI, were used to determine the weight 0.3, optimal values of thresholds, and other parameters in these equations.

Once a node determines that a neighbor, node A, is compromised, it will send out three reports about the compromise to the base station. The reports are all identical with fields for the reporter, the node being reported, and the type of misbehavior that triggered the report. From that point on, the reporter will continue to record information from node A, but will not perform any more calculations to detect anomalies unless instructed otherwise by the base station.

B. Centralized Component

The centralized component, running on the base station, is responsible for deciding if a reported node is actually compromised, based on data from other nodes, or if the reporter made a mistake. If the reported node is compromised, the base station will alert the user and perform any recovery procedures, such as ignoring all of its messages. However, if the reported node is determined to not be compromised, the base station will tell the reporter(s) to treat it as non-compromised and continue monitoring.

For all new neighbor reports, the user is alerted. If an actual node was added to the network, the user can instruct the network that it is not malicious. For other cases, the base station will process data based on reports from other nodes. In order to make a decision about whether a reported node is compromised, ComSen uses a *beta reputation system* [4].

It has been shown that beta reputation systems are able to accurately detect misbehavior based on numerous reports and lower the high false positive rates in strict detection systems; because the system takes history into account, in order to successfully hide a compromised $nodeA$, on average 72% of $nodeA's$ neighbors have to be compromised (in contrast to the 33-50% for other approaches like majority voting) [11], [16], [26]. Beta reputation systems are an extension that uses probability density functions to combine feedback from multiple sources and determine a reputation rating, or rating of how trustworthy the subject node is. In our case, the reputation rating corresponds to a decision of compromised or not if it is past a threshold value.

In beta reputation systems, the probability, ρ, that a reported event is accurate given two parameters α and β is the beta distribution, $f(\rho|\alpha,\beta)$, which can be expressed using the gamma function Γ as:

$$f(\rho|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)+\Gamma(\beta)}\rho^{\alpha-1}(1-\rho)^{\beta-1}, \quad 0 \le \rho \le 1, \alpha > 0, \beta > 0 \qquad (3)$$

We chose the parameters, α and β, to be the weighted sum of past reports for the reported node and the number of compromised nodes within two-hops of the reported node. These allow the network topology and past reports to impact the final decision about whether a reported node is compromised. Initial baseline values are determined during the setup period.

The base station has knowledge of every node's neighbor, which may have been gathered during the setup period. A report about node A being compromised by node B has a higher probability of being correct the more of node A's neighbors report it, the longer its history of reported, and the more compromised nodes there are near it.

 ISBN: 978-1-61208-209-7

On the other hand, if the probability is too low, then the reporter, node B, may be compromised and launching a slander attack against node A. In which case, the base station will instruct other nodes that node B is compromised, alert the user, and launch recovery procedures.

This approach prevents adversaries from using ComSen to attack the network without being detected. If a compromised node impersonates the base station, nodes closer to the base station on the routing path will detect messages coming from the wrong direction and alert the base station. Thus, an adversary cannot impersonate the base station without being immediately detected.

ComSen is also not vulnerable to slander attacks. Suppose a compromised node, node C, wants to slander one of its neighbors, node D. As previously mentioned in Section IV-A2, impersonating other nodes will be detected by neighbors and nodes cannot influence each other. The base station knows node C's neighbors, so reporting a non-neighbor will also result in detection. The only possible slander attack is for node C to influence the base station by sending reports of compromised neighbors. However, without supporting reports from node D's neighbors, the base station will not consider node D as compromised. Moreover, the slander attack may only cause node C to be considered compromised.

V. Implementation

We implemented ComSen using TinyOS [27]. There were two key issues that may be applicable to other implementations of ComSen.

First, recall that every node has a misbehavior buffer for each of its monitored neighbors. The format and size of that buffer is implementation-dependent (i.e., depends on the needed accuracy and performance of the WSN). Nevertheless, the reports in the buffer must consist of the following fields: time of alert, type of misbehavior, and its source.

Second, there may be a high memory overhead for the buffers need to monitor all neighbors. The overhead grows as a quadratic function of the number of immediate neighbors, which is not scalable for high density networks. To address this, ComSen nodes can select a subset of neighbors to monitor. The selection can be random or come from other protocols. For example, in random pairwise key distribution protocols [7], [9], there are a number of keys generated before deployment and each node is given a random key. After deployment, there is a probability that two neighboring nodes will have compatible keys and be able to communicate. Depending on the density of the network, it's possible to control the average number of neighbors that each node can communicate with it by adjusting the total number of keys. As we will show in later sections, the performance of ComSen is maximized after a certain number of monitored neighbors, so it's unnecessary to monitor all neighbors in dense networks.

VI. ComSen's Performance

In order to analyze the performance of ComSen, we ran a series of experiments using SenSec [29], an evaluation tool which allow us to simulate and analyze various attacks on WSNs running real applications. The experiments provided quantitative analysis of the effectiveness of ComSen with different parameters.

We used the standard metrics of performance for detection systems [8], [10], [15]:

- **Detection rate:** This metric is the percentage of actual compromises that were detected by the system. However, even with a 100% detection rate, the accuracy of the system cannot be determined without considering false positives.
- **False positives:** It is possible for legitimate nodes to be reported as compromised. These reports are known as false positives. The detection rate is not inversely proportional to the false positive rate. For example, a system can have a 100% detection rate and still have a 99% false positive rate. Systems with a high percentage of false positives are inaccurate because the majority of reported compromises are false.
- **Detection time:** Detection mechanisms require time to collect and process collected data before making a decision on whether a node is compromised. Detection time is the interval during which a compromised node remains undetected.

Section VI-A discusses the performance of the distributed component. The overall performance of ComSen (i.e., both its distributed and its centralized component) is discussed in Section VI-B.

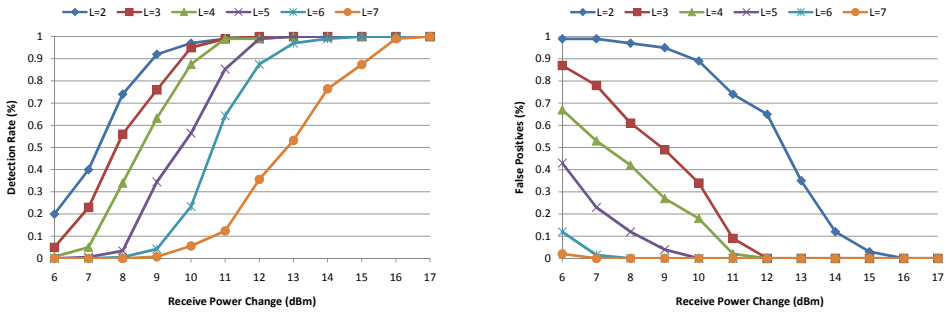
A. Distributed Component's Performance

In modeling the compromise of nodes in a network, we used a gradient-based model proposed by Chen et. al. [3]. This model is based on the observation that compromises exhibit spatial locality. For example, if adversaries are compromising nodes while walking from one node to the next, then the chances of a node being compromised is higher if they're closer to a compromised node. Thus, the probability that a node is compromised forms a gradient, with higher probability for nodes closer to compromised nodes and vice versa.

Our network topology consists of 100 simulated motes randomly deployed over a 100m x 100m area. The motes have radios with transmission powers of 5 dBm and are running common sensor applications that taken sensor readings every 1s and routing them to a base station at a random edge of the network. Tree routing and CSMA are used. There is a setup period. At some random time, after the setup period, a random node in the simulated WSN is compromised every 10 simulated minutes and launches a series of attacks against the network. The attacks launched by compromised nodes are all those provided by SenSec, such as replay, sybil, wormhole, pulse-delay, selective forwarding, etc. (i.e., the performance is independent of the attack). A real TinyOS application that takes sensor readings every 0.1 s is run for each node in the simulation. As we vary the parameters in the detection algorithms, we analyze their effect on ComSen's performance. Every experiment consisted of 50 runs, each lasting 1 simulated hour.

Figures 6(a) and 6(b) show the results of the experiments where we evaluated the performance of the detection algorithm

 ISBN: 978-1-61208-209-7



(a) Detection rate (b) False positives

Fig. 6. Performance of detection algorithms based on receive power change

that uses receive power at various receive power changes and packet buffer lengths (L). The threshold value is 1 dBm. For the first L packets, we use an unchanging 5dBm for the transmission power, while the receive power is simulated based on the physical topology. Then the power level of the transmission power is increased.

From the results, we can see that smaller packet buffers require smaller changes in the receive power to detect a compromise; a buffer of length 2 can achieve a 95% detection rate with the smallest change in receive power. However, the tradeoff is a higher false positive rate; a buffer of length 2 has a 95% false positive rate. These results are explained by our using past data to establish the baseline and a smaller buffer means that the algorithm is more sensitive to small changes whether they're caused by compromises or transient changes in the environment.

We found that the detection times remain constant for each buffer length, regardless of the change in receive power. The graphs were omitted for brevity, but they demonstrate that the buffer length is the deciding factor in the algorithm's sensitivity.

These results also show that, it is possible to achieve a false positive ratio of less than 10% with small packet buffers (i.e., lengths in the single digits). Thus, the memory overhead imposed by ComSen is low.

The experiments on the sensor reading detection algorithm produced similar results, which are omitted for brevity.

For the algorithm that uses send rate, we used the buffer length (L) of 6, and a sublength (L_2) of 2. Figures 7(a) and 7(b) show the performance of this algorithm as we vary the receive power, by some percentage, and the threshold value K. The results from this experiment mirror those of the previous experiment: lower values, for threshold and buffer length, lead to more responsive algorithms that offer better detection rates at the cost of higher false positive rates. For example, if K is 1.02, then a 90% detection rate is achieved with a 30% increase in the send rate, but the false positive rate is 97%. Once again, the detection times are dependent only on K and remain constant as the send rate changes.

The experiments on the receive rate detection algorithm produced similar results, which are omitted for brevity.

These experiments are meant to analyze the possible performance of the detection algorithms deployed on every node in the WSN. The actual parameters should be adjusted according to application security requirements. However, these results show that the algorithms are capable of detecting compromised nodes with detection of over 98% and false alarm of under 5%.

Of course, the performance varies greatly under different environments, because a single node with limited resources cannot achieve high accuracy under all conditions. The role of the detection algorithms is to notify the base station of *possible* compromises. The base station will compensate for the limitations of motes by aggregating reported compromises from multiple sources and decide if a report is correct.

B. Overall Performance

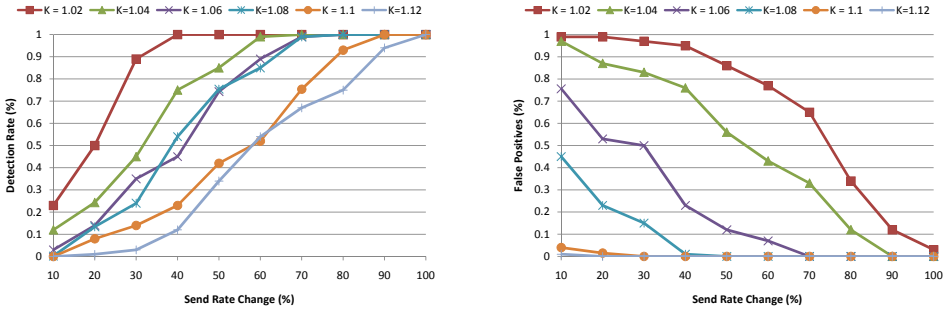
We ran several experiments to provide quantitative analysis of the performance of ComSen. The experimental setup is identical to the one used in Section VI-A, where 100 nodes, running real TinyOS applications, are randomly deployed.

For our first experiment, we wanted to evaluate the effects of increasing the number of neighbors monitoring each other. Furthermore, most detection systems perform poorly (detection rates of less than 50% and false positive ratios of more than 90%) in lossy environments [28]. So, we varied both parameters and analyzed the results.

Figure 8 shows the results of the ComSen's performance evaluation at various loss rates and number of monitoring neighbors. Figures 8(c) and 8(b) shows that the algorithm can compensate for high loss rates with more neighbors monitoring each other. At 30% loss rate, a 99% detection rate and a 2% false positive ratio can be achieved if each node monitors an average of 9 neighbors. Higher loss rates affect the detection rate more than the false positive ratio, because lost reports make real compromises look like transient errors. However, in most cases, the compromise is detected as future malicious behavior is reported, so the detection time is higher for larger loss rates (Figure 8(c)).

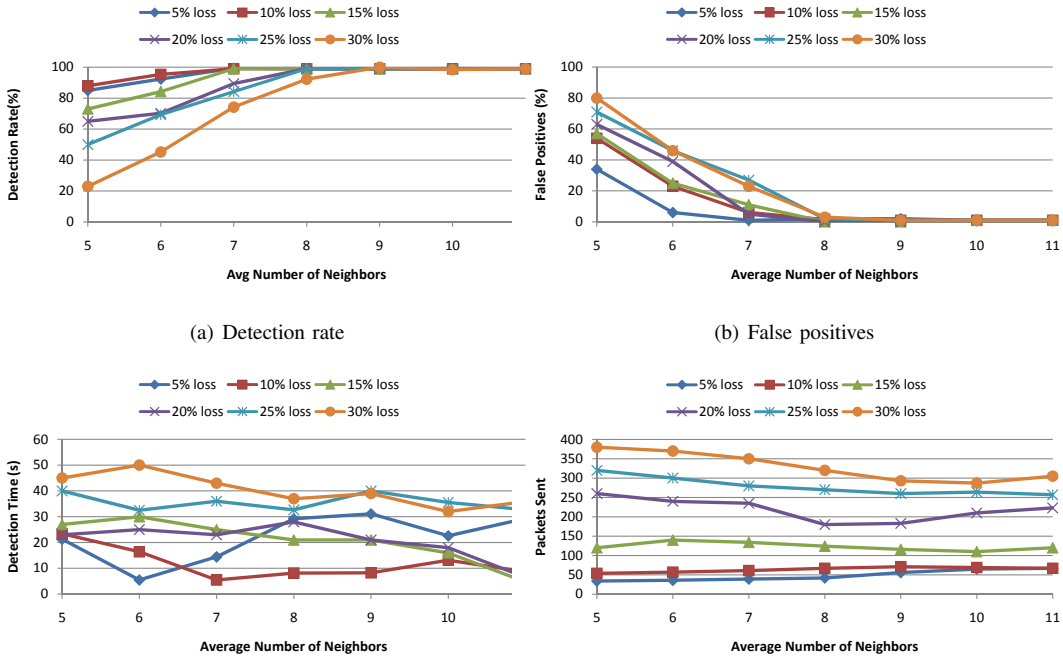
We also measured the communication overhead, the number of packets sent related to the operation of ComSen, which is shown in Figure 8(d). As expected, at higher loss rates, more packets are sent due to retransmissions. However, increasing the number of monitoring neighbors does not increase the number of packets sent. In the majority of cases, the number of packets sent does not change significantly and, in some cases, the number of packets sent actually decreases 5% for every neighbor added. Closer inspection shows that when the number of monitored neighbors is increased, more neighbors are sending reports based on the same misbehavior, which increases the communication overhead. However, having more generated

 ISBN: 978-1-61208-209-7



(a) Detection rate (b) False positives

Fig. 7. Performance of detection algorithms based on send rate change



(a) Detection rate (b) False positives

(c) Detection time (d) Communication Overhead

Fig. 8. Performance of ComSen at various loss rates and number of monitoring neighbors

reports allows the base station to detect compromised nodes faster (i.e., the detection time is lower) when some of the reports are lost. Thus, the future misbehavior of compromised nodes won't generate reports, which lowers the communication overhead. In conditions where the loss rate is more than 15%, the net result is a decreased or unchanged communication overhead with each additional neighbor.

We measured the energy consumption of ComSen. The energy consumption of the radio accounts for the majority, of the total energy consumption, which is consistent with previous results [1], [18], [25]. Since the total energy consumption is proportional to the communication overhead, the graphs were omitted for brevity.

For our second experiment, we evaluated the effects of network density on the previous results. We increased the overall network density from 100 nodes to up to 10,000 nodes. Our results showed that density has no significant effect on ComSen's performance. We omit the results for brevity, but Figure 8 does not change significantly with network density.

We also ran experiments on the computational and memory overheads, which are omitted for brevity. As expected, they are $<1\%$ for motes, because the distributed component only performs simple computations and requires monitoring a constant number of neighbors.

These results demonstrate that ComSen can provide accurate detection of compromised nodes and is scalable to large networks. Although similar systems offer comparable performance with no loss, with 30% loss rate, their best detection rate drops to 14% and the false positive ratio becomes 99% [28]. However, ComSen can provide a 99% detection rate and a 2% false positive ratio at 30% loss rate. Moreover, its overhead is not significantly increased for denser or larger networks, scaling up to thousands of nodes

VII. Conclusion and Future Work

In WSNs, compromised nodes can undermine the integrity of data by sending false data reports, injecting false data, and disrupting transmissions. Since cryptographic solutions are not sufficient to prevent these attacks, we proposed ComSen, a system for detecting compromised nodes in WSNs.

 ISBN: 978-1-61208-209-7

This paper has presented the design of a novel and reliable detection system that is not vulnerable to slander attacks, where malicious nodes use the detection algorithm to launch attacks. Through a series of experiments, we showed that ComSen can provide detection rates of 99% and false positive ratios of less than 2% in environments with loss rates of 30%. ComSen can run on most WSNs because it uses common application features (sensor readings, receive power, send rate, and receive rate) and can adjust its detection behavior if the sensor application doesn't have periodic transmissions or lacks inter-node communication. It has low memory, computation, and communication overhead that allows it to scale to networks of over thousands of nodes.

Possible directions for future work include creating a response system and adding a challenge system. ComSen provides a means of identifying compromised nodes in the network but not how to respond to such a compromise. The most basic approach would be to isolate the offending node, but that may not be appropriate for all scenarios. Furthermore, ComSen does offer high accuracy in identifying compromised nodes, but once ComSen determines that a node is compromised, it could challenge the node to prove that it isn't compromised, improving the accuracy further.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, Aug 2002.

[2] C. Castelluccia, A. Francillon, D. Perito, and C. Soriente. On the difficulty of software-based attestation of embedded devices. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 400–409, New York, NY, USA, 2009. ACM.

[3] X. Chen, K. Makki, K. Yen, and N. Pissinou. Node compromise modeling and its applications in sensor networks. In *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, pages 575 –582, July 2007.

[4] B. E. Commerce, A. Jsang, and R. Ismail. The beta reputation system. In *In Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

[5] M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei. A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '07, pages 80–89, New York, NY, USA, 2007. ACM.

[6] M. Demirbas and Y. Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, WOWMOM '06, pages 564–570, Washington, DC, USA, 2006. IEEE Computer Society.

[7] R. Di Pietro, L. V. Mancini, and A. Mei. Random key-assignment for secure wireless sensor networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, SASN '03, pages 62–71, New York, NY, USA, 2003. ACM.

[8] D. Djenouri, L. Khelladi, and A. Badache. A survey of security issues in mobile ad hoc and sensor networks. *Communications Surveys Tutorials, IEEE*, 7(4):2 – 28, 2005.

[9] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8:228–258, May 2005.

[10] X. Du and H.-H. Chen. Security in wireless sensor networks. *Wireless Communications, IEEE*, 15(4):60 –66, Aug 2008.

[11] S. Ganeriwal and M. B. Srivastava. Reputation-based framework for high integrity sensor networks. In *In SASN 04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 66–77. ACM Press, 2004.

[12] Y.-C. Hu, A. Perrig, and D. Johnson. Packet leashes: a defense against wormhole attacks in wireless networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1976 – 1986 vol.3, Mar 2003.

[13] N. James, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 259–268, New York, NY, USA, 2004. ACM.

[14] I. Khalil, S. Bagchi, and N. B. Shroff. Liteworp: A lightweight countermeasure for the wormhole attack in multihop wireless networks. *Dependable Systems and Networks, International Conference on*, 0:612–621, 2005.

[15] C. Krau, M. Schneider, and C. Eckert. On handling insider attacks in wireless sensor networks. *Information Security Technical Report*, 13(3):165 – 172, 2008.

[16] F. Li and J. Wu. Mobility reduces uncertainty in manets. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1946 –1954, May 2007.

[17] Mica2. http://www.xbow.com/, June 2012.

[18] M. J. Miller and N. H. Vaidya. A mac protocol to reduce sensor network energy consumption using a wakeup radio. *IEEE Transactions on Mobile Computing*, 4:228–242, 2005.

[19] K. Okeya and T. Iwata. Side channel attacks on message authentication codes. In R. Molva, G. Tsudik, and D. Westhoff, editors, *Security and Privacy in Ad-hoc and Sensor Networks*, volume 3813 of *Lecture Notes in Computer Science*, pages 205–217. Springer Berlin / Heidelberg, 2005.

[20] I. Onat and A. Miri. An intrusion detection system for wireless sensor networks. In *Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005)*, volume 3, pages 253 – 259 Vol. 3, Aug 2005.

[21] B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensor networks. In *Security and Privacy, 2005 IEEE Symposium on*, pages 49 – 63, May 2005.

[22] A. Seshadri, M. Luk, and A. Perrig. Sake: Software attestation for key establishment in sensor networks. In *DCOSS '08: Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems*, pages 372–385, Berlin, Heidelberg, 2008. Springer-Verlag.

[23] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla. Scuba: Secure code update by attestation in sensor networks. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 85–94, New York, NY, USA, 2006. ACM.

[24] A. Seshadri, A. Perrig, L. V. Doorn, and P. Khosla. Swatt: Software-based attestation for embedded devices. In *In Proceedings of the IEEE Symposium on Security and Privacy*, 2004.

[25] V. Shnayder, M. Hempstead, B.-r. Chen, G. W. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 188–200, New York, NY, USA, 2004. ACM.

[26] Y. L. Sun, Z. Han, W. Yu, and K. J. R. Liu. A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks. In *IEEE INFOCOM*, pages 230–236, 2006.

[27] TinyOS. http://www.tinyos.net, June 2012.

[28] Y.-T. Wang. *Accurate and Scalable Security Evaluation Framework for Wireless Sensor Networks*. PhD thesis, University of California, Los Angeles, Los Angeles, CA, USA, Nov. 2011.

[29] Y.-T. Wang and R. Bagrodia. Sensec: A scalable and accurate framework for wireless sensor network security evaluation. In *SN '11: The Fourth International ICDCS Workshop on Sensor Networks*, pages 230–239, 2011.

[30] Y. Zhang, J. Yang, W. Li, L. Wang, and L. Jin. An authentication scheme for locating compromised sensor nodes in wsns. *J. Netw. Comput. Appl.*, 33:50–62, Jan 2010.

 ISBN: 978-1-61208-209-7