University of Technology Darmstadt, Germany
Department of Computer Science
IT Security Group

Seminar: Sicherheit in drahtlosen Sensornetzen
WS 2005/06

Topic:

Detecting Misbehavior in Wireless Sensor Networks

(March 5, 2006)

Supervisor:

Dipl.-Inform. Christoph Krauß

Autor: Nils Knappmeier
Studiengang: Informatik

Contents

Chapter 1

Introduction

This chapter gives a brief introduction to the topic of misbehavior in wireless sensor networks and describes the structure of this work. Section 1.1 gives a very brief introduction to wireless sensor networks and their characteristics, section 1.2 briefly displays the common security problems in sensor networks and section 1.3 outlines the reasons to deal with misbehavior detection.

Chapter 2 then gives a broad overview of several detection and reaction mechanisms.

Chapter 3 presents an intrusion detection system for wireless sensor networks. It is worth being described in more detail, because it deals with a broad range of attack forms, while other works primarily concentrate on a single attack.

Chapter 4 presents a system to filter false information. This work is important, because it expects a form of attack that is not mentioned in chapter 3. It also presents a way of dealing with intrusion automatically rather than only detecting it.

My own remarks and ideas will be presented in chapter 5 and chapter 6 will draw a conclusion of the presented material.

1.1 Wireless Sensor Networks

Wireless sensor networks usually consist of a number of sensor nodes being deployed in a potentially hostile environment. These nodes do not have a lot of computing power and run on battery, thus their energy resources are highly constraint. Each node has two functions

1. Perform measurements using the integrated sensors and send them towards a sink, which gathers the measurements of the whole network.

2. Forward measurements of other sensors towards the sink.

In order to conserve energy, the third task is to aggregate measurements from different sensors and forward the resulting measurement to the sink. The main problems of sensor networks are the limited computing, storage and energy resources.

1.2 Security

These constraints have an impact on the security functions of a sensor network: Public key cryptography is difficult to apply in this area, because the computer would need to perform complex algorithms. This creates a need for key-distribution and authentication schemes that are based on symmetric cryptography only. On the other hand, a potential attacker is not subject to the same resource limitations as the sensors and may use his superior computing or transmission power to attack the network. This is said only to present a brief picture of the area of security in wireless sensor networks.

1.3 The need for misbehavior detection

Intrusion detection is also a topic in traditional wired networks. For sensor networks, it is an even more important concern, because the sensors are potentially positioned in a hostile area. If an attacker has physical access to the nodes, he can access the memory directly and extract secret cryptographic information. Even if the network uses encrypted and authenticated communication, he can use the stolen keys to authenticate himself as a sensor node and perform the same kinds of attacks that are possible on an unsecured sensor network. It is then important that the network is able to detect and isolate nodes that have been compromised.

This does not render cryptography useless. For example, the intrusion detection system presented in chapter 3 assumes that all sensor nodes are uniquely identifiably. This can only be achieved by some kind of cryptographic authentication. However, there is also a need for additional mechanisms to identify misbehaving nodes and ensure the network's functionality if a node has been compromised.

Chapter 2

Misbehavior detection, reaction and tolerance

This chapter gives an overview of several works on the topic of misbehavior detection. Section 2.1 summarizes the attacks mentioned in these works, their characteristics and relation to conventional network failures. Section 2.2 briefly describes methods of data aquisition and analysis in order to identify potential attacks.

Different works in the area of misbehavior detection in sensor networks have different focuses on the topic and different approaches to deal with misbehavior. Although it is never explicitly stated, it is often implied, that the detection of misbehavior is not enough. It is also important to handle misbehaving nodes, once they are detected. The reason is, that sensor nodes have a higher level of autonomy than traditional networks and even mobile ad-hoc networks. Once deployed, there is no user sitting next to the node, who could be warned of an intrusion taking place. The alarm message has to be routed over the same medium, i.e. the air, and is subject to the same attacks as the "normal" communication in the network. Therefore, section 2.3 gives an overview of different ways of dealing with misbehavior.

2.1 Attack opportunities

As said in section 1.3 we assume that the attacker has the opportunity to attack the network using stolen cryptographical data from compromised nodes. The following attacks are mentioned in [1], [2] and [3]:

- **Message delay:** A node does not forward a message immediately but only after a certain delay time.
- **Wormhole:** An attacker creates a faster connection through the network using more radio power. It then controls the connection that most other nodes will use and can run other kinds of attacks on packets passing through (see figure 2.1).
- **Message repetition:** An attacker forwards the same message multiple times, in order to perform a replay attack or an exhaustion attack.
- **Jamming:** An attacker floods the network with packets and causes collisions in order to make communication between nodes impossible. This can be mistaken with message collisions as they happen occasionally in sensor networks.

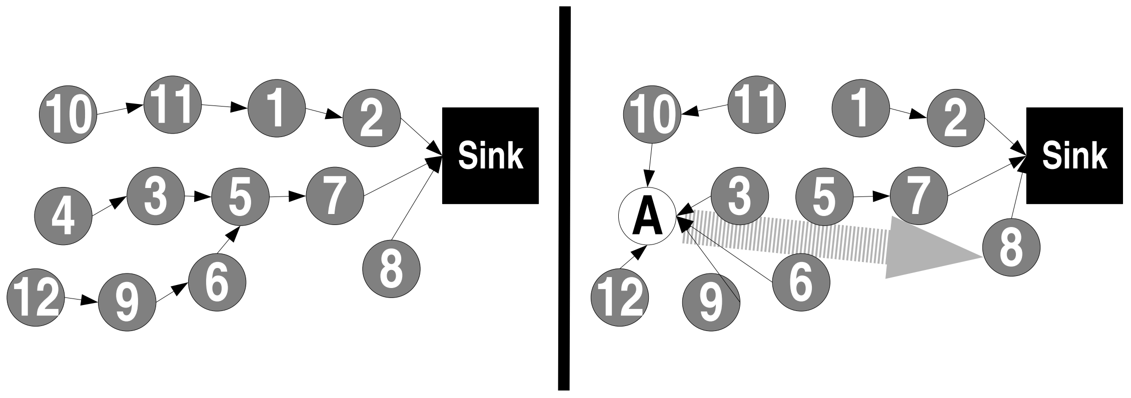

Figure 2.1: Attacker performing a wormhole attack. The left picture shows the routing tree before the attack. In the right picture, the attacker controls the connection of the nodes 3,6,9,10,11 and 12 to the sink.

- **Data alteration:** An attacker forwards modified data packets. This also can occasionally happen as a network failure.
- **Blackhole and selective forwarding:** An attacker claims to be able to forward packets, but then simply drops them completely or on a selective basis. This attack can be mistaken for the occasional message loss that happens in a sensor network.
- **Exhaustion attack:** An attacker continuously sends messages in order to raise the power consumption of the forwarding nodes.
- **Injection of false data:** An attacker injects fabricated measurements into the network.

2.2 Detecting misbehavior

There are different approaches to detect misbehaving nodes. [1] uses redundant measurements from multiple sensors to confirm reports created by one node (see chapter 4). The data forwarding behavior can be verified by nearby nodes listening in promiscuous mode. This can either happen on selected nodes (as in [2]) or on every single node in the network (as in [4]).

A method that is only applicable to black hole and selective forwarding attacks is presented in [3], where every sensor node sends acknowledgements back to the sink, when a packet is received. That way, the sink can identify locations, where packets get lost and make preparations to avoid these black holes.

2.3 Reacting on misbehavior

While the intrusion detection system in [2] makes no effort to deal with intrusion automatically, other works implement methods that make the network robust and more tolerant to misbehaving nodes: The system for “location-centric isolation of misbehavior” in [3] uses a blacklist-embedding in packet headers to route packets around areas with misbehaving nodes. The “statistical en-route filtering” in [1] attempts to drop measurements that have not been verified by a number of nodes on the route to the sink. The reputation framework in [4] computes a reputation value for neighboring nodes, representing the probability that information received from this node is authentic.

In the following two chapters, the intrusion detection system [2] and the “statistical en-route filterning” [1] will be presented in more detail.

Chapter 3

An intrusion detection system for wireless sensor networks

This chapter summarizes the architecture and results presented in [2] as an example for a method to gather and analyze information for intrusion detection in wireless sensor networks.

Section 3.1 provides some common information about intrusion detection. Section 3.2 presents the general architecture of the IDS for sensor networks. Section 3.3 describes the rule setup that was used to detect anomalies. Finally, the sections 3.4 and 3.5 show the setup and the results of the evaluation.

3.1 Intrusion detection in wired networks

Traditional intrusion detection systems (IDS) can be either net-based or host-based. A net-based IDS monitors the network traffic for unusual behavior while a host-based IDS analyzes the files and programs on the computer itself in order to find out if it has been infiltrated.

Another way to define intrustion detection systems is by their data analysis approach. A behavior-based IDS tries to learn the *normal* traffic and detect deviations of this normal behavior. A pattern-based IDS recognizes specific attack patterns. The advantages and disadvantages are discussed in more detail in [5]

3.2 Architecture of the IDS for sensor networks

In sensor networks, since a potential attacker has physical access to the sensor nodes, a host-based intrusion detection system would only be able to recognize the intrusion when the attacker is already capable of reprogramming the whole node or extracting the secret keys. That is why the presented IDS for sensor networks is implemented in a net-based fashion: Some of the sensor nodes use promiscuous listening to monitor and analyze the network traffic between the nearby sensors.

It is pattern-based in the sense that a static set of rules is used to detect malicious behavior. Since some attacks have similar indicators as occasional network failures (see section 2.1), the system only raises an alarm if the number of rule applications exceeds a threshold. This threshold is determined during a learning phase directly after deploying the sensors. Thus partly, the system also uses the behavioral paradigm. It is important to mention that the IDS assumes that

- once deployed, nodes don't change their position anymore and that
- every node is uniquely identifiable.

3.3 Detection algorithm

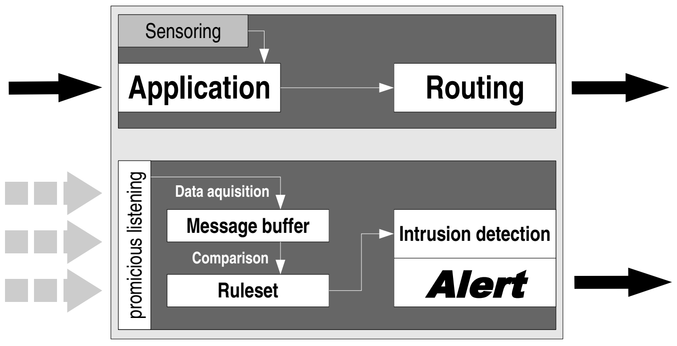

Figure 3.1: Architecture of a monitor node ([2])

The architecture of a monitor node is shown in figure 3.1. The detection algorithm is divided in three steps, *data acquisition*, *rule application* and *intrusion detection*. In the *first step*, the node is listening in promiscuous mode and stores the received messages into a message buffer. In the *second step*, each of the following rules is applied to the stored messages:

- **Interval rule:** Limits are defined for the minimal and maximal allowed time between two consecutive messages of a sending node. For a time below these limits, an exhaustion attack is assumed[1].
- **Retransmission rule:** The monitor checks whether the nodes in its listening range forward received messages. If a node receives a message, but does not forward it, a blackhole or a selective forwarding attack is assumed.
- **Integrity rule:** The monitor checks for each node whether the payload of the received message matches the payload of the forwarded message. Otherwise a data alteration attack is assumed.
- **Delay rule:** An upper limit is defined for the time in which a node must forward a received message. Otherwise a message delay attack is assumed.

[1]At this point, [2] also assumes a message negligence attack for a time above these limits. This attack is never mentioned again and thus omitted here.

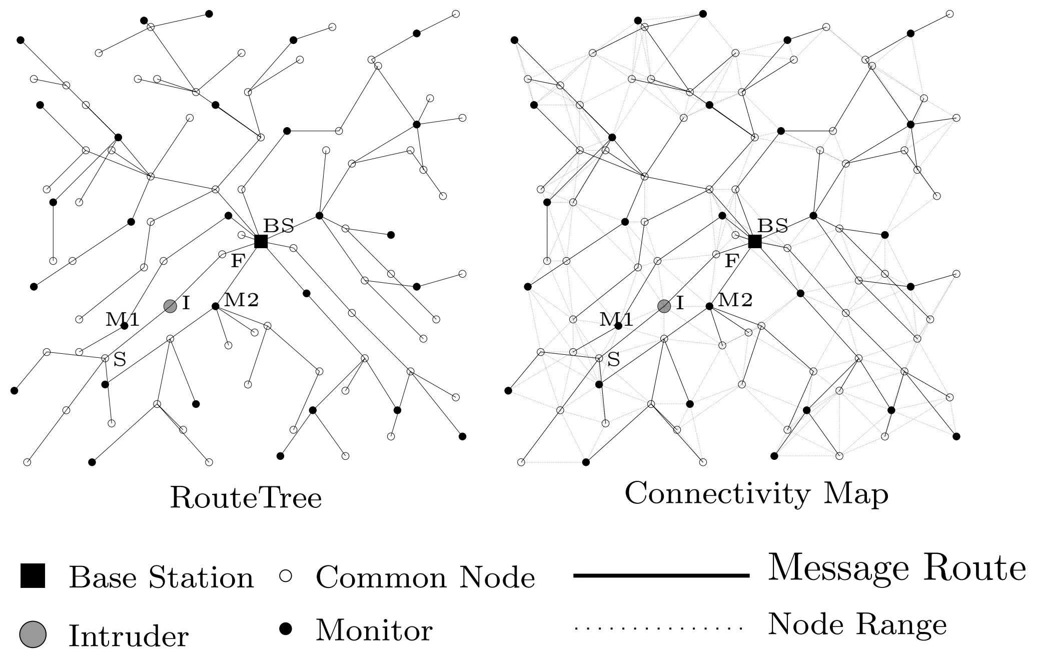

Figure 3.2: Route tree and connectivity map of an example IDS setup ([2])

- **Repetition rule:** An upper limit is defined for the number of times that a node may send the same message.
- **Radio transmission range rule:** A message received by the monitor node must originate from a node within the radio transmission range. Otherwise a wormhole attack is assumed. The nodes within transmission range can be identified during the learning phase.
- **Jamming rule:** The number of message collisions must remain under a certain threshold. Otherwise, a jamming attack is assumed.

Each time a rule applies to the events in the network, a counter is incremented and if the average number of failures exceeds the threshold determined during the learning phase, an alarm is raised in the *third step* of the algorithm.

3.4 Simulation setup

The system has been tested in a simulator with a setup of 100 sensor nodes, 28 of which also acted as a monitor node. Figure 3.2 shows the routing tree and the connectivity tree of an example setup. The simulation was divided in 10000 iterations. It began with a 1000 iteration learning-period and continued with the intruder being in turn idle for 700 iterations and then attacking for 200 cycles. Network failures[2]

[2]Data alteration, message loss, message collision

were simulated with a probability of 10% (20% in another simulation). In each simulation, the compromised node was attacking with one specific attack[3]. The size of the message buffer was set to 30, 60, 100, 200 and 400 messages.

For each attack, the number of false positives and the percentage of detected attacks (detection rate) was measured.

3.5 Simulation results

This section will briefly display the most important simulation results. A complete analysis can be found in [2].

3.5.1 The size of the message buffer

An important result is the relation between message buffer size and the efficiency of the IDS. The message buffer size is a measure of the resource overhead needed, so this relation actually represents the trade-off between resources and efficiency.

- **False positives:** A result of the simulations was, that the number of false positives greatly depends on the size of the message buffer. Smaller buffer sizes create more false positives. The given explanation was, that the influence of the simulated network failures is higher with small buffer size, because the variance is bigger over a small set of samples.
- **Delay attack:** The recognition rate of the delay attack was almost proportional to the buffer size. This seems logical, since messages have to be saved for a long time in order to recognize this attack.
- **Data alteration:** The data alteration attack had such a high number of false positives for small buffer sizes, that the detection rate was hardly significant. The detection rate was higher for smaller buffer sizes. The explanation in the paper is: "This happens because on larger buffers the generated failures of an attacker do not take the averages of occasional network failures too high" [2, page 21]. There is a statement[4] about this topic in section 5.1.

3.5.2 Reliably detectable attacks

Independently of the size of the message buffer, the detection rate of the **message repetition**, **wormhole** and **blackhole** attack were consistently over 90%. The detection of the **selective forwarding** attack depended only a little on the buffer

[3]Message delay, message repetition, wormhole, jamming, data alteration, blackhole or selective forwarding

[4]I admit that I do not understand the explanation, which is why I provide some personal remarks.

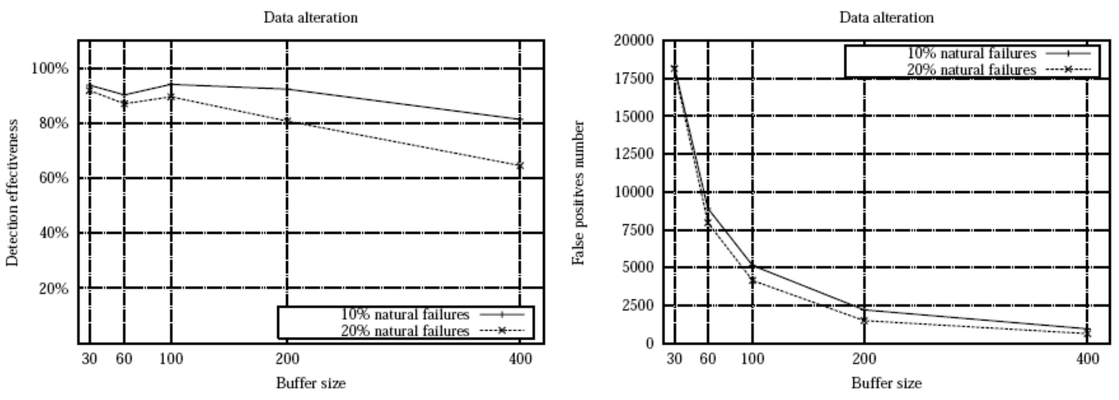

Figure 3.3: Detection effectiveness and false positives for the data alteration attack ([2])

size, with a detection rate of over 80% for buffer sizes bigger than 60. Figure 3.3 shows this relation.

3.5.3 Falsely identified attacks

In some cases, attacks were mistakenly reported as a different attack. The delay attack was frequently reported as a blackhole attack, for small buffer sizes. The jamming attack was in some cases mistakenly identified as a blackhole attack, when another node was unable to send any data due to the jamming, but the monitor node could not detect the jamming directly.

Some of the attacks[5] can only be detected by monitor node, if it is within radio range of the attacker *and* the previous node in the routing tree. It is suggested to distribute the monitors so that for every edge in the routing tree at least one monitor can observe both the sender and the receiver.

3.5.4 Energy consumption

The energy consumption of sensor and monitor nodes was simulated, but not described in greater detail in the paper. In the simulation, the energy need of each **transmission** and **reception** of messages and for **listening** on the network was approximated. The energy values of each action was computed for 36 byte messages as $Q_{transmission} = 0.48375\frac{mJ}{message}$, $Q_{reception} = 0.1575\frac{mJ}{message}$ and $Q_{listening} = 0.00875\frac{mJ}{message}$. While "reception" means the full processing of a message, "listening" includes only the verification of the receiver address in the message header of each message.

The main result was, that monitor nodes consume more energy than standard sensor nodes and that the energy consumption is generally higher near the sink.

[5]Data alteration, delay, blackhole, selective forwarding

Chapter 4

Statistical en-route filtering of injected false data

This chapter presents the "statistical en-route filtering of injected false data" described in [1].

The goal of en-route filtering is to identify fabricated reports created by compromised nodes. This identification should at least happen at the sink, but in order to reduce the energy consumption of the network, false packets should be dropped as early as possible on the routing path.

The general idea of this work is to distribute the nodes in the network in a density, that a number of sensors can always confirm the readings made by another sensor.

Only symmetric cryptography is used, but a special key-distribution scheme allows nodes to falsify reports. This scheme is described in section 4.1. The algorithms to generate reports and to identify forgeries are presented in 4.2 and 4.3 respectively. Finally, section 4.4 presents the methods and results of the evaluation.

4.1 The key distribution scheme

A key distribution scheme has been introduced along with the en-route filtering system. The goal of this scheme is to give every node enough information to identify a false report with a certain probability, yet not enough information to let an attacker create false reports by compromising one or two nodes.

In the first step, a set of N keys $\{K_1, \dots, K_N\}$ for Message Authentication Codes is generated, each of which is assigned a unique index number i. The set is then partitioned into m disjoint subsets (categories) of size n. In order for the algorithm to work, the category of a key K_i must be derivable from the index number i.

Before deploying the network, k keys are stored into each sensor node. The keys are distributed among the nodes such that each node contains keys from exactly one category.

4.2 Report generation

When a stimulus is detected by some nodes, the node with the strongest reading is elected as the center of stimulus (CoS). It creates a report $R = (pos, time, type)$

where *pos* is the position of the event, *time* is the time when the event happened and *type* describes what kind of event was observed. This information is sent to the neighboring nodes, which verify the reading based on their own observations. Each node returns the index i of a random key in their storage and a Message Authentication Code (MAC) $M_i = \text{MAC}(R, K_i)$ back to the CoS. The CoS then selects t MACs from distinct categories, where t is a predefined constant and attaches them to the report. The final report is $(R, i_1, M_{i_1}, i_2, M_{i_2}, \ldots, i_T, M_{i_T})$.

In [1], bloom filters are used to reduce the overhead caused by the attached MACs by merging multiple MACs into one value. A method is presented there but ommited here due to space limitations.

4.3 En-route filtering

There are three conditions that a report has to fulfill in order to be considered valid:

1. t different index numbers are attached to the report.
2. Each index number i is from a distinct category.
3. Each Message Authenticiation Code M_i was produced by the key K_i and the report R.

Every node en-route to the sink performs these tests as far as it is able to do so. This means: It verifies the first and the second condition and drops every report that does not apply to these rules. Furthermore, it checks the third condition *if* the index number of a key in the storage matches the index number of a MAC attached to the report. There is a probability that this is the case due to the key distribution scheme. If the node can falsify one of the MACs, the report is dropped. If all MACs are either valid or not verifiable, the report is assumed to be valid and forwarded towards the sink.

In the end, the sink is able to verify all MACs attached to the report, because it has access to all keys.

4.4 Evaluation

The en-route filtering has been evaluated theoretically and simulated in a setup with 340 nodes and a distance of 100 hops between sink and the location of the stimulus. The results of the simulation were consistent with the theoretical analysis, which is why they are omitted here.

The goal of the analysis was to predict the detection rate and the energy savings of the en-route filtering.

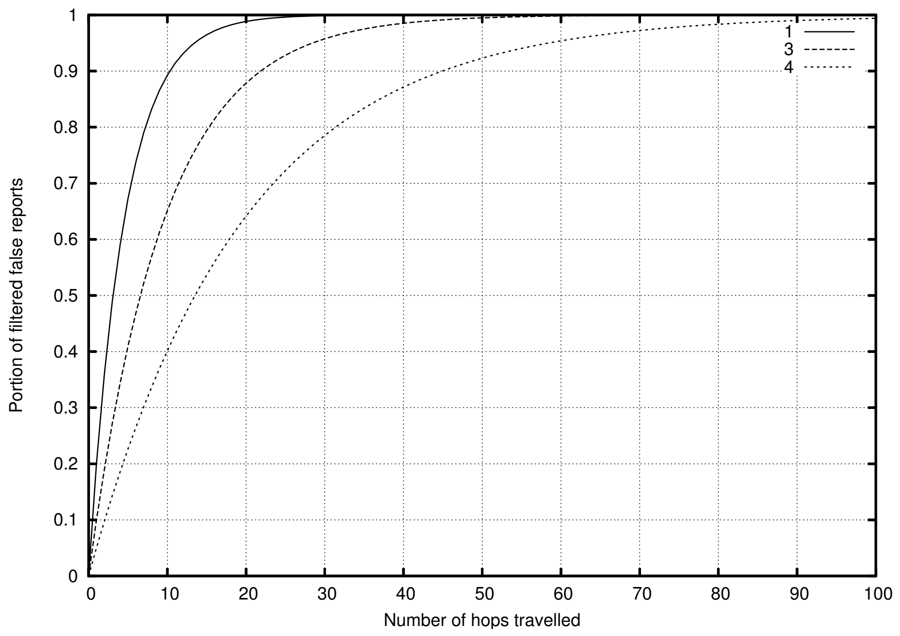

Figure 4.1: Number of identified false report after n hops, for 1,3 and 4 compromised categories [1]

4.4.1 Detection rate

The theoretical evaluation was based on the following assumptions: Let m be the number of categories, n the number of keys per category and $N = n \cdot m$ the number of total keys. Every node stores k keys from one category and t different MACs are necessary to provide a valid report. Assuming the attacker has compromised keys from m_c categories and tries to forge a report by guessing the remaining keys, the probability, that a randomly chosen node stores one of the guessed keys, thus is able to falsify the report, is

$$p_0 = \frac{t - m_c}{m} \cdot \frac{k}{n} = \frac{k \cdot (t - m_c)}{N}$$

This formula is derived from the probability that the verifying node stores keys from the category of the guessed key ($\frac{t-m_c}{m}$) and the probablity that a the key for a randomly chosen index number from this category is stored in the node ($\frac{k}{n}$). En-route to the sink, the probability that a report travels x hops before being identified as a fake is

$$p_x = 1 - (1 - p_0)^x$$

Figure 4.1 shows the percentage of dropped reports after n hops if the attacker was able to compromise 1,3 or 4 categories. The other parameters are chosen as $t = 5$ MACs, $m = 10$ categories, $n = 100\frac{keys}{category}$ and $k = 50\frac{keys}{node}$. With one category compromised by the attacker, 90% of the false reports were dropped within 10 hops and even if 4 categories are compromised, almost all false reports are dropped withing 100 hops.

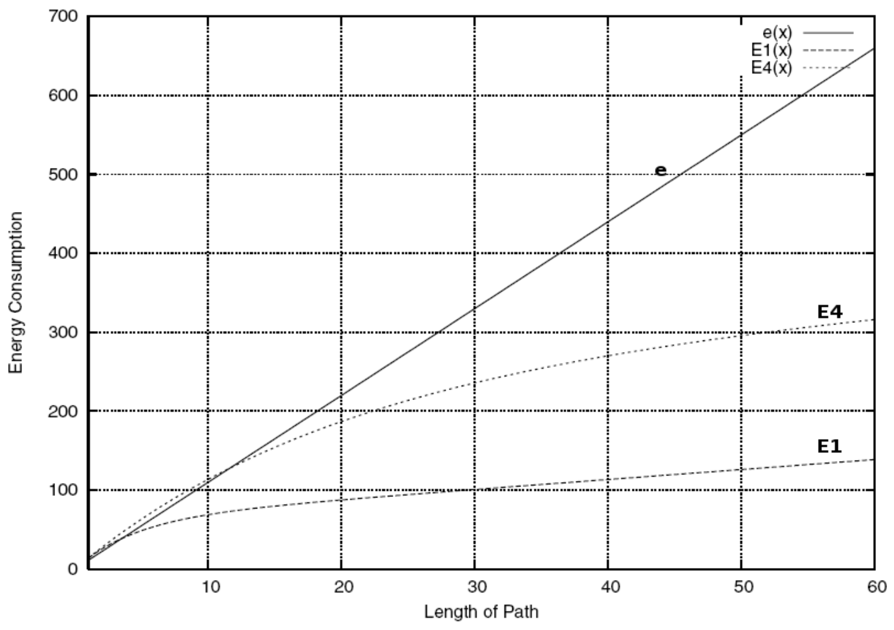

Figure 4.2: Average energy consumption of each report without en-route filtering (e), with one compromised category($E1$) and with four compromised categories ($E4$) [1]

4.4.2 Energy consumption

The energy consumption is a an important measure because the goal of the en-route filtering was, to save energy by dropping false reports early. It was computed based on the assumption that the transmission and reception of a packet cost 60mW and 12mW respectively and that the number of forged reports were 10 times the number of valid reports. The number of packets received and transmitted by each node was computed using the formulas given above. Figure 4.2 shows the energy savings due to the en-route filtering. The length of the attached key indices and MACs, in form of a bloom filter, has been taken into account as well. The exact formula is omitted here because bloom filters are not explained in detail in this work.

Chapter 5

Assessment and further ideas

In this chapter, I will present my own comments and remarks about both described systems. Additionally, I will outline ideas for enhancing and integrating both systems.

5.1 Remarks on the IDS

This section describes some remarks and possible improvements to the intrusion detection system presented in chapter 3.

Feasibility of the IDS The discussion after my seminar presentation led to the following concern: The simulation of the IDS showed that the system works well only if the message buffer is large enought. From the given diagrams (see figure 3.3), a minimum size of 200 messages can be suggested in order to reduce the number of false positives to an acceptable level. Assuming a message size of 36 bytes[1], the memory usage of the IDS is $200 \cdot 36 = 7200$ bytes. The available memory size of, e.g. the MICA2 sensor node[6] is 4096 bytes. This shows that, although the IDS is an intersting approach, it is not feasible on this kind of node.

Detecting data alteration The detection of data alteration attacks was rather unsuccessful in the IDS due to the high number of false positives. Unfortunately, the authors did not state, whether the simulated nodes were using any error-correction schemes in their link layer protocol. However, an error-correction based on CRC and resend-requests still leaves the connection from the supervised node to the supervising monitor open to transmission errors. A monitor cannot tell the node to resend a packet, because it is not the actual receiver. It is not explicitly said, whether the monitor performed a CRC check prior to verifying the forwarded message, but the high number of false positives indicates that it was not done.

I suggest that the monitor discard messages that do not pass a CRC check. An additionally applied forward-error-correction scheme could enable the monitor to correct at least a percentage of the messages before checking the payload for data alteration.

[1]This size is used for some applications in TinyOS[7]

Infiltrating the IDS The attacker can as easily compromise a monitor node as a normal sensor node. This possibility is completely ignored in the paper. Using the identity of a monitor, an attacker could raise false alarms as well as prevent alarm messages from reaching the sink. This could be partly avoided, if the monitor nodes use a system as described in chapter 4.

Instant infiltration The authors in [2] mention that the attacker may need some time to infiltrate a node and extract the cryptographic information needed for an attack. This constraint is, however, never mentioned again. Assuming that the node cannot send any data, while being compromised, that should be detected by the monitor nodes as well, maybe as a blackhole attack.

Dedicated monitor nodes It may be a good idea to deploy dedicated monitor nodes among the sensors, which are not involved in the routing of normal messages. Since these monitors would not send any data unless an intrusion takes place, they would be harder to find by a potential attacker.

Furthermore, the energy consumption of a monitor node might even be less than the consumption of a normal node, if it does not participate in the routing. The monitor will only send data, if an intrusion is detected and according to section 3.5.4, sending consumes three times the energy of receiving. Still, this would have to be tested, because monitors have to *receive* all message instead of just *listening* to them.

This is again a trade-off between cost and efficiency, since the monitor nodes represent new resources, thus a cost factor, but cannot be used for the actual goal of the network, which is measuring data.

5.2 Key distribution of the “en-route filtering”

The key distribution presented in chapter 4 suggests that not all the keys of a partition have to be stored in a node. The reason is, that fewer keys will be put at risk if the node is compromised. From the formulas and the description of the system however, it does not matter how many keys of each category are compromised in order to create a false report. One key of each category is enough to make the report seem valid. If no key revocation system is implemented, which is not the case in [1], a category size of *one* with *one* key per node would be the most efficient use of resources with no reduction of the detection efficiency.

Another point has to be noted: Even though the en-route filtering makes forging reports harder than in an unprotected network, a constant number t of compromised nodes suffice to break the protection. This t is a constant parameter that does not depend on the size of the network. Furthermore, in order for the system to work, nodes with keys from t different categories have to be present near any location, where an event might occur. An attacker can simply listen in on the network traffic, locate those nodes and compromise the keys necessary to forge a report[2]. It would be desirable not to have a constant threshold like this.

5.3 Combining en-route filtering with intrusion detection

The two systems presented in chapter 3 and 4 complement one another in the sense that the only attack not detected by the intrusion detection system, injection of false data, is handled by the en-route filtering system. It seems reasonable to combine both ideas in order to create a system that is more tolerant to intrusions.

Barring the feasibility concern against the IDS (see section 5.1), the only problem is that the IDS could not implement rules to detect black hole and selective forwarding attacks due to the following reasons:

- The **message loss rule** would occur every time a message is dropped because of an invalid MAC
- The monitor node might not be able to distinguish a correctly dropped message from a blackhole attack, because it does not neccessarily have all the keys to verify the MACs of the drop message.
- If every monitor would store all existing keys, the MACs could be verified, but compromising a single monitor node would be sufficient for an attacker to create false reports.

5.4 Unaddressed problems

None of the papers mentioned in this work handles the detection of wrong aggregation behavior. This is a more complex and energy consuming task than detecting the presented attacks, but it is an important topic in my eyes.

Another interesting topic would be a method to hide intrusion alarms from the attacker. If an attacker has physically compromised a node then he can eavesdrop on intrusion reports and find out if the compromisation has been detected. A way to raise a silent alarm that only the sink can detect would be desirable.

Finally, none of the presented systems is able to handle mobile sensor nodes, which is important e.g. in wildlife surveillance.

[2]This flaw was the result of a discussion my seminar presentation

Chapter 6

Conclusion

In this work, two different approaches for misbehavior detection have been presented. The requirement for both systems was the successfull detection of misbehavior in sensor networks. Both approaches have fulfilled this requirement in their given scenario, but there are still many open problems (see chapter 5).

It has to be noted that the scenarios were very expicitly defined within certain constraints. For example, both systems were simulated with non-mobile nodes in an environment with no obstacles. The problems of re-deployment and key-revocation have been ignored. The IDS does not deal with the topic of encrypted communication.

Other scenarios are thinkable, like the sink polling for new data, continous measurings, mobile nodes or the detection of unauthorized persons trying to access the network to perform measurings. These kind of scnearios are not dealt with.

This should not be seen as a flaw in the presented works. Other works also specialized on a certain scenario. The wide range of applications for sensor networks makes it impossible to present a general solution for all scenarios.

The conclusion is: The presented works introduce some good ideas, but there are many details that have to be worked on yet.

Bibliography

[1] Statistical En-route Filtering of Injected False Data in Sensor Networks, Fan Ye, Haiyun Luo, Songwu Lu, Lixia Zhang, UCLA Computer Science Departement, Los Angeles

[2] Decentralized Intrusion Detection in Wireless Sensor Networks, Ana Paula R. da Silva, Marcelo H. T. Martins, Bruno P. S. Rocha, Antonio A. F. Loureiro, Linnyer B. Ruiz, Hao Chi Wong, October 2005, Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks Q2SWinet '05

[3] Location-centric Isolation of Misbehavior and Trust Routing in Energy-constrained Sensor Networks, Sapon Tanachaiwiwat, Pinalkumar Dave, Rohan Bhindwale, Ahmed Helmy, University of Southern California, Los Angeles

[4] Reputation-based Framework for High Integrity Sensor Networks, Saurabh Ganeriwal and Mani B. Srivastava, University of California Los Angeles

[5] Claudia Eckert. IT Sicherheit - Konzepte, Verfahren, Protokolle. R.Oldenbourg Verlag, 3.Edition, October 2004

[6] Crossbow, MICA2 Wireless Measurement System Data Sheet, `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf`

[7] `http://www.tinyos.net`