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
{anapaula, marcelo, bpontes, loureiro, linnyer, hcwong}@dcc.ufmg.br
Dept of Computer Science
Federal Univ of Minas Gerais
Belo Horizonte, MG, Brazil

ABSTRACT

Wireless sensor networks (WSNs) have many potential applications. Furthermore, in many scenarios WSNs are of interest to adversaries and they become susceptible to some types of attacks since they are deployed in open and unprotected environments and are constituted of cheap small devices. Preventive mechanisms can be applied to protect WSNs against some types of attacks. However, there are some attacks for which there is no known prevention methods. For these cases, it is necessary to use some mechanism of intrusion detection. Besides preventing the intruder from causing damages to the network, the intrusion detection system (IDS) can acquire information related to the attack techniques, helping in the development of prevention systems. In this work we propose an IDS that fits the demands and restrictions of WSNs. Simulation results reveal that the proposed IDS is efficient and accurate in detecting different kinds of simulated attacks.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*; H.1.0 [**Models and Principles**]: General; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms

Design, Security

Keywords

Wireless Sensor Networks, Security, Intrusion detection

Q2SWinet'05, October 13, 2005, Montreal, Quebec, Canada.

1. INTRODUCTION

Wireless sensor networks (WSNs) constitute a new paradigm of ambient monitoring with many potential applications. Typically formed by thousand of nodes of small dimension, they use ad-hoc communication and have scarce resources regarding energy, bandwidth, processing capacity and storage. WSNs are typically designed to gather data in inhospitable places and might be involved in critical applications. Wealth environment mapping and enemy's movement monitoring in a battlefield are some examples of critical applications they are used for. In these applications, WSNs are of interest to adversaries.

WSNs are susceptible to some types of attacks [10, 25] since they are deployed in open and unprotected environments and are constituted of cheap small devices. Preventive mechanisms can be applied to protect WSNs against some types of attacks [9, 18]. However, there are some attacks for which there is no known prevention methods, such as wormhole [10, 5]. Moreover, there are no guarantees that the preventive methods will be able to hold the intruders. For these cases, it is necessary to use some mechanism of intrusion detection. Besides preventing the intruder from causing damages to the network, the intrusion detection system (IDS) can acquire information related to the attack techniques, helping in the development of prevention systems.

Intrusion detection poses many challenges to WSNs, mainly due to the lack of resources. Besides, methods developed to be used in traditional networks cannot be applied directly to WSNs, since they demand resources not available in sensor networks. WSNs are typically application oriented, which means they are designed to have very specific characteristics according to the target application. The intrusion detection assumes that the normal system behavior is different from the behavior of a system under attack. The several possible WSN configurations make difficult the definition of the "usual" or "expected" system behavior.

Since common nodes are designed to be cheap and small, they do not have enough hardware resources. Thus, the available memory may not be sufficient to create a detection log file. Moreover, a sensor node is designed to be disposed after being used by the application and it makes difficult to recover a log file due to the possible dangerous environment in which the network was deployed. The software stored in the node must be designed to save as much energy as possible in order to extend the network lifetime. Finally, another challenge to the design of an IDS is the frequent

failures of sensor nodes when compared to processing entities found in wired networks. Given all these characteristics, it is important to detect the intrusions in real time. In this way, we could hold the intruder and minimize the application damages.

In this work, our goal is to study the problem of intrusion detection on WSNs and present an intrusion detection system that fits the demands and restrictions of those networks. Our main contributions are: the proposal of a decentralized IDS model tied to the WSN restrictions and peculiarities; a high-level methodology to construct a specific IDS to a target WSN with well defined applications; the assessment of the IDS efficiency and accuracy in detecting seven different kinds of simulated attacks; the evaluation of the utilization costs concerning energy consumption and memory utilization; and the development of an IDS simulator that is able to represent the main characteristics of WSNs.

The proposed IDS is based on the inference of the network behavior obtained from the analysis of events detected by a monitor node, i.e, the node that implements the IDS system. The only events we consider are: data message listened to by the monitor that is not addressed to it, and message collision when the monitor tries to send a message.

The rest of this paper is organized as follows. Section 2 briefly describes the related work. Section 3 presents the rules proposed for our IDS. Section 4 discusses the proposed algorithm used in the IDS. Section 5 presents some design issues associated with the IDS. Section 6 presents and discusses several simulation results using the proposed IDS. Finally, Section 7 presents the conclusion for this work.

2. RELATED WORK

An important aspect of the broad area of security is intrusion detection. Many solutions have been proposed to traditional networks [8, 7, 20, 17, 6, 12], but restrictions of WSN resources make direct application of those solutions inviable.

Ad-hoc networks have similarities with WSN. These networks also have severe resource restrictions, although they are not as restrictive as WSNs. There are solutions presented to intrusion detection for ad-hoc networks, but too little has been made in relation to WSNs.

In [2], an IDS model for ad-hoc networks is presented following the behavioral paradigm. The IDS is decentralized and detection is made by clusters. A technique to safely elect the responsible node for monitoring each cycle was developed. This solution is expensive, thus being inadequate to a WSN.

In [5], it is proposed a method for detecting wormhole attacks in ad-hoc networks by evaluation of the time spent on the transmission of packets between nodes in the network, and by node authentication. This work proposes two protocols: *Slot Authenticate MAC* and *TIK*. Both need synchronization of the network trusted time. Since it is hard to keep nodes synchronized in a WSN, we have not considered this premise. In [19], it is shown how to detect attacks such as wormhole and HELLO flood in WSNs by comparing the power of the received signal with the power of the observed signal in the network. For the specific wormhole attack, we have used a simpler strategy based on the network topology, in which it is sufficient for the monitor node to know the identities of its neighbors. The strategy proposed in [19] still can be used as one of the rules of our system, in case of the nodes of the target network being able to measure the power of the received signal. Our work proposes a wider solution, capable of detecting several types of intruders and attacks.

In [14], it is introduced the idea of a watchdog for ad-hoc networks in order to improve the detection of mischievous nodes. It uses a technique called pathrater to help routing protocols to avoid those nodes. In this work, we have used a similar idea. As we will see, the monitor node watches its neighbors to know what each one of them will do with the messages it receives from another neighbor. If the neighbor of the monitor nodes changes, delays, replicates, or simply keeps the message that should be retransmitted, the monitor counts a failure. This technique is also used to detect other types of attacks.

In [4], it is presented a routing protocol that tries to keep the network functioning even on the presence of intruders. It is a fault-tolerant solution based on route redundancy. However, many of the attacks found on literature cannot be tolerated, which motivates the development of an IDS adequate to WSNs.

3. RULES AND DEFINITIONS

The following steps must be taken to construct an appropriate IDS to a target WSN: (1) pre-select, from the available set of rules, those that can be used to monitor the features defined by the designer; (2) compare the information required by the pre-selected rules with the information available at the target network to select rules definitively; and (3) set the parameters of the selected rules with the values of the design definitions. In the following, we present the definition of the available rules:

Interval rule: a failure is raised if the time past between the reception of two consecutive messages is larger or smaller than the allowed limits. Two attacks that will probably be detected by this rule are the negligence attack, in which the intruder does not send data messages generated by a tampered node, and the exhaustion attack, in which the intruder increments the message sending rate in order to increase the energy consumption of its neighbors.

Retransmission rule: the monitor listens to a message, pertaining to one of its neighbors as its next hop, and expects that this node will forward the received message, which does not happen. Two types of attacks that can be detected by this rule are the blackhole and the selective forwarding attack. In both of them, the intruder suppresses some or all messages that were supposed to be retransmitted, preventing them from reaching their final destination in the network.

Integrity rule: the message payload must be the same along the path from its origin to a destination, considering that in the retransmission process there is no data fusion or aggregation by other sensor nodes. Attacks where the intruder modifies the contents of a received message can be detected by this rule.

Delay rule: the retransmission of a message by a monitor's neighbor must occur before a defined timeout. Otherwise, an attack will be detected.

Repetition rule: the same message can be retransmitted by the same neighbor only a limited number of times. This rule can detect an attack where the intruder sends the same message several times, thus promoting a denial of service attack.

Radio transmission range: all messages listened to by the monitor must be originated (previous hop) from one of its neighbors. Attacks like wormhole and helloflood, where the intruder sends messages to a far located node using a more powerful radio, can be detected by this rule.

Jamming rule: the number of collisions associated with a message sent by the monitor must be lower than the expected number in the network. The jamming attack, where a node introduces noise into the network to disturb the communication channel, can be detected by this rule.

In the retransmission, integrity, delay, repetition and interval rules, the monitor suspects about its neighbors. In this way, besides detecting an attack, we have both the address and location of the intruder.

4. PROPOSED ALGORITHM

The proposed algorithm was divided into the following phases: **Phase 1 – Data acquisition**: in this phase, messages are collected in a promiscuous mode and the important information is filtered before being stored, for subsequent analysis. **Phase 2 – Rule application**: this is the processing phase, when the rules are applied to the stored data. If the message analysis fails the tests being applied, a failure is raised. **Phase 3 – Intrusion detection**: this is the analysis phase when the number of raised failures is compared to the expected amount of occasional failures in the network. If the former is higher than the latter, a intrusion detection is raised.

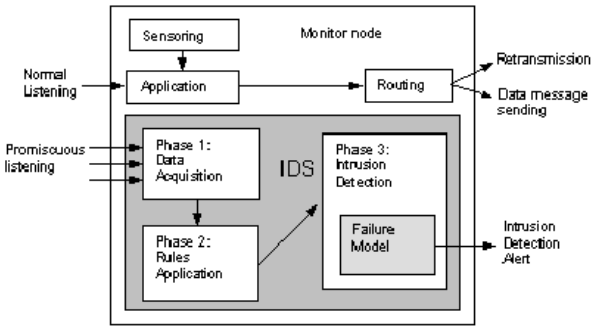

Figure 1: Detection phases

Figure 1 shows the architecture of a monitor node. This node runs the common node functions, like sensoring and data message sending and retransmitting, in addition to the IDS functions. The IDS has three modules, each one being responsible for a phase.

Phase 1 – Data Acquisition

In this phase, messages are listened to in promiscuous mode by the monitor mode and the important information is filtered and stored for subsequent analysis. Important information includes message fields that might be useful to the rule application phase. Thus, we use less memory and less processing time, saving energy. Messages to which no rules can be applied are not stored.

Data extracted from the messages are stored in an array data structure and discarded after a given period of time or when there is no space left in memory.

Phase 2 – Rule Application

In this phase, each entry in the array data structure is evaluated according to a sequence of rules specific to each message type. If a message fails in one of the rules, a failure counter is incremented. At this moment, the message can be discarded and no other rule will be applied to it. We have adopted this strategy due to the fact that WSNs have severe resource restrictions. This strategy makes sense since the first failure already gives us an indication of an abnormal behavior in the network. This strategy also reduces the detection latency. We have here a trade off between the accuracy, processing cost, and running time.

Rules are applied to the stored data in increasing order of complexity. After being tested against all rules without failing any of them, the message is discarded.

Phase 3 – Intrusion Detection

In order to implement an IDS which is capable of, in most cases, distinguishing occasional network failures from attack instances promoted by intruders, we have proposed a solution in which a monitor node can infer the purpose of a suspect node participating on the network, since these failures are similar to the attacks. The following solution addresses the issues raised by attacks such as data alteration, message negligence, blackhole, selective forwarding, and jamming.

In our model, an attack is raised only if, after counting all network failures detected by the monitor node during the analysis of messages transmitted on its neighborhood in a round, its number is greater than an expected value. This number is calculated dynamically by the monitor node, using a failure history for each node in its neighborhood. An average of the number of failures that have occurred since the deployment of the sensor nodes is kept and updated every time the IDS is activated. The history update takes place only if the number of failures for that round is close to the cumulative value kept by the monitor. In this case, the value of the round failure and the previous cumulative value are combined and form a new cumulative value.

This technique introduces the idea of a deviation tolerance. Although occasional failures may happen during each round of message capture by the monitor nodes, its number is not known beforehand. By determining the variance bounds for it, an IDS can raise an attack indication whenever these limits are reached. In other words, an attack indication is only signaled by the monitor node when an abnormal behavior occurs with a frequency higher than expected.

Considering that the failure expectancy takes time to stabilize, a large number of false positives will appear at the beginning of the network life cycle. To avoid this, a learning stage has been introduced, in which a monitor node does not consider, during a certain period, any abnormal event in order to prevent an attack indication from being mistakenly signaled while the average has not settled down. This learning stage must not last long, or else the damage promoted by possible intruders on the network can be overwhelming. Algorithm 1 summarizes the technique described above.

5. IDS SIMULATOR

There are some simulators developed or adapted for WSNs, such as SensorSim [16], TOSSIM [13], and PowerTOSSIM [23]. Unfortunately, none of them are appropriate for our purpose. We have developed our own WSN simula-

Algorithm 1 Comparing round-failure average with failure history

```
for all neighbors do
  for all failure types do
    if round-failure value > cumulative value then
      signal attack indication
    else
      update cumulative value by combining it with round-failure value
    end if
  end for
end for
```

tor [15]. This simulator has been implemented in C++ and designed with three goals in mind: performance, modularity, and extensibility. We have implemented a discrete event model, in which the analysis objects (base station, common nodes, monitors and intruder) keep their states during the simulation until the occurrence of some event such as receiving or sending a message, the occurrence of sensoring and the activation of an attack. Network sensoring events are generated randomly and nodes are not synchronized, as an attempt to approximate the simulator to the behavior of a real network. Our simulator is composed of the following modules: network, message, sensor node, monitor node, intruder node, events and attacks generator, IDS and stats collector. The network module is responsible for the message exchange between modules in such way that it can simulate the functioning of a real WSN.

We have considered four types of nodes: common node, monitor, intruder and base station. The *common node* has sensor and router functions. As a sensor, it collects sensoring data, and sends it to the base station. As a router, it retransmits all messages directed to the base station. The *monitor* is responsible for monitoring its neighbors looking for intruders. By doing this, the node keeps its radio in a promiscuous mode, storing relevant information and processing it according to selected rules. This node also executes the sensor/router functions since it is a common node where an IDS was installed. The *intruder* node switches between a common node behavior and an intruder behavior. The intruder behavior depends on the considered attack. The intruder can spend from 1% to 100% of its time performing an attack. In the context of these experiments the *base station* is only the destination of all data messages.

Only attacks over data messages were taken into account. In our proposed WSN model, we have considered three types of occasional network failures and eight types of intruder attacks. Occasional network failures are listed as follows:

1. **Data alteration**: occurs when the message payload is changed to a different value from the original.
2. **Message loss**: a message sent by a node is lost while being transmitted. Its origin is not aware of the loss.
3. **Message collision**: a message is lost while being transmitted and its origin detects the loss due to a collision.

The simulator was designed to perform experiments with configuration and routing messages, but they were postponed for future work. The data message considered here has the following fields: next hop, message type, previous hop, origin, final destination, sequence number, and data.

We have simulated a plan and fixed network [21], with random node distribution. These nodes are uniquely identified and have a fixed radio range. In the following, we describe the network features considered and the associated attacks and rules.

The messages follow the routing tree made from the distributed algorithm Propagation of Information [22] in a multihop way. Many nodes have to retransmit the message and if this node is tampered, it can perform the selective forwarding and blackhole attacks. We use here the retransmission rule.

The data message comprises the sensor reading. Since we do not have any kind of data fusion or aggregation, the message received by the common node has to be transmitted with no payload alteration. We can detect the intruder that modifies the message using the integrity rule.

We do not have receiving acknowledgment or retransmission features, then a message sent more than one time is interpreted as an attack by the repetition rule.

A node can only receive messages from neighbors. The radio range rule can detect the wormhole and helloflood attacks.

We consider a timeout for the node to transmit a message based on the simulator. If an intruder delays the message, it can be detected by the delay rule.

We do not consider message collisions on the network, so any collision is interpreted by the jamming rule as an attack attempt.

In our simulator, occasional network failures follow a probabilistic model, which takes place every time a message is sent by a node. Occasional network failures can be mistakenly detected as an attack instance by the IDS, generating a false positive. The following attacks were considered in our work:

- **Message delay, repetition and wormhole**: since these attacks are not related to failures such as message loss or data alteration, they cannot be mistaken for the network failures mentioned above.
- **Jamming**: a transmitter, tuned to the same frequency as the receiving equipment can, with enough power, override any signal at the receiver, preventing it from receiving any messages. This type of attack can be mistaken for occasional message collisions by the monitor node.
- **Data alteration**: this attack is equivalent to the occasional network failure of the same name, but in this case, the alteration happens on behalf of the intruder node.
- **Message negligence, blackhole and selective Forwarding**: an intruder node ignores messages which should be sent or forwarded. These attacks can be mistaken for occasional message losses by the monitor node. Conversely, the occasional faults can also be misidentified as attack instances.

We have simulated a network comprised of 100 nodes randomly distributed as shown in Figure 2. Data messages are sent from 40 to 40 simulation cycles. In Figure 2 we have two representations of the same network: the routing tree and the connectivity map. In both maps, a solid line connecting two nodes means that they communicate directly to each other. In the connectivity map, a dotted line connecting two lines means that they are neighbors. M_1 and M_2 nodes are the monitors in our focus. F and S nodes are, respectively, the parent and the child of the intruder in the routing tree.

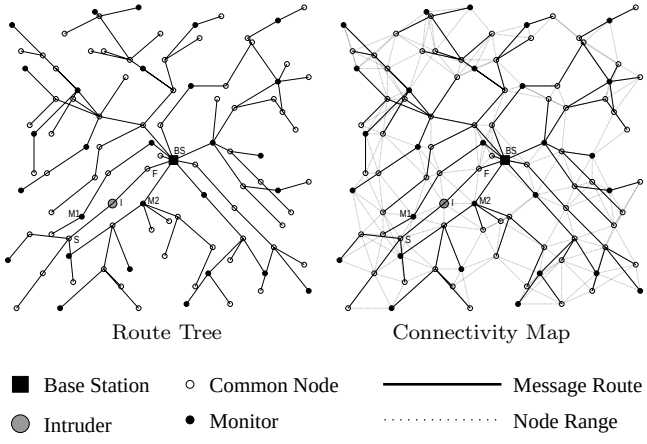

Figure 2: Routing tree and connectivity map

All mentioned rules were used in all simulations run. We have used 28 monitors distributed in such a way that all common nodes could be monitored. In Figure 2, M_1 and M_2 nodes are the only monitors who are neighbors to the intruder and, thus, the only ones that can directly observe its behavior. Even though many common nodes are monitored by more than one monitor, the point of view of each one is not the same. M_1 node, for example, can listen to messages from the intruder's child but it cannot listen to messages from the intruder's parent. On the other hand, M_2 node can listen to messages from the intruder's parent but cannot listen to messages from the intruder's child. According to the considered attack, one of these monitors can detect the intrusion and the other cannot.

6. SIMULATION RESULTS

In this section, we present the simulation results for the IDS proposed in this work.

6.1 Initial Considerations

The goal of our experiments is to evaluate the proposed IDS. In particular, the amount of raised detections and false positives. From the monitor's point of view, the simulation time is divided in slices. Each time slice starts when the array is empty and begins to store messages captured in promiscuous mode. The time slice finishes when the array is completely full and the message processing can be triggered. The length of the array defines the size of the time slice when the monitor is hearing in promiscuous mode, and, thus, defines the amount of messages that can be related to each other in order to find an intruder. There is a trade off between the storage cost and the detection efficiency. If the array is small and, consequently, its storage cost, the time slices will be small and the message sequence break will be large, which implies a worse detection efficiency. In order to evaluate this trade off, we have considered array sizes of 30, 60, 100, 200 and 400 messages, for each kind of attack.

All simulations runs for 10000 iterations. There was always only one intruder (I) performing one single type of attack, which was varied in each simulation. The network has a "learning period" of 1000 iterations (10% of its total lifetime) in which there is no attack and the monitors cannot generate any attack report either. After that, the intruder begins its attack cycle, which consists of 700 iterations resting and then 200 iterations attacking. The IDS has a tolerance of 10%, which means that a failure ratio calculated from a message buffer can be 10% larger than the node accumulated failure average, without generating an attack report. Other factor that has been varied is the probability of occasional network failures, which changed between 10% and 20%.

The results are presented in the form of correct detections and false positives. The correct detections show the percentage detection levels of the IDS closest to the intruder (M_1). The false positives show an absolute number of false positives collected by M_1 and M_2 (both close to the intruder) over the 10000 iterations.

6.2 Detection Effectiveness

One aspect that is common to all attacks is that using a small buffer size results in a larger number of false positives. This happens because small buffers produce less accurate failure averages, since less messages are stored between each buffer processing. This lack of accuracy leads the IDS to generate false positives when the number of occasional failures is a bit higher than its usual rating. The variation of buffer sizes also impacts on detection level, as well as the variation of occasional failures probability do. We will discuss separately the results for each attack.

6.2.1 Repetition, Delay and Wormhole

The effectiveness and the number of false positives for these attacks are depicted in Figures 3 to 8, respectively. These attacks are not mistaken with any kind of occasional failure, so neither the detection levels nor the number of false positives are influenced by the probability of occasional failures. The detection of delay attacks is directly proportional to buffer sizes, since with smaller buffer sizes the IDS receives the delayed message at the beginning of the buffer more often. With a buffer size of 30 messages, the detection effectiveness was only 30%. For buffer sizes greater than 30, detection was between 60% and 90%. Detection of the repetition attack proved more successful, with detection levels never below 90%. In wormhole attacks, detection was always 100% due to the rule to detect this kind of attack.

The number of false positives for these attacks were lower than the ones for other attacks, since in this case there is no confusion with occasional network failures. In all attacks, a buffer size of at least 100 guarantees no more than 1000 false positives for two IDSs after 10000 iterations, which means that each IDS has on average 1 false positive for each 20 iterations.

In the repetition attack, the monitor M_2 generated false positives, reporting as the intruder of the repetition attack the parent node F of the real intruder. This happens because our network has no kind of suppression of repeated messages, so node F simply forwards the repeated messages it receives.

In the delay attack the higher number of false negatives happens because the monitor can listen to some of the messages the intruder will delay at the end of the buffer. Thus, the delayed message will only be listened to at the beginning of the next buffer. False positives of this attack happen when the monitor listen to a message before the delayed one, the timeout of retransmission is reached, but the delayed message is not listened to in the same time segment. This way, a blackhole report is generated, thus being a false positive.

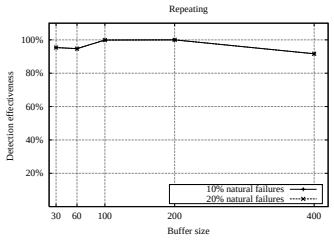

Figure 3: Detection effectiveness of repetition attack

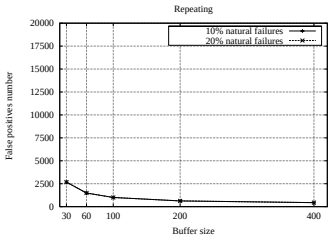

Figure 4: Number of false positives of repetition attack

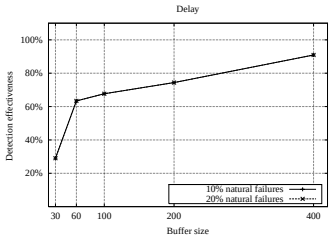

Figure 5: Detection effectiveness of delay attack

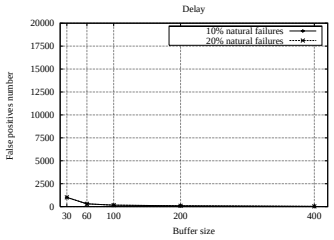

Figure 6: Number of false positives of delay attack

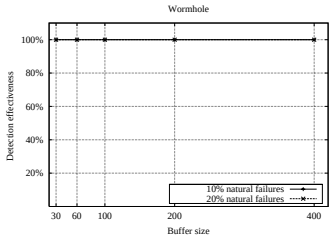

Figure 7: Detection effectiveness of wormhole attack

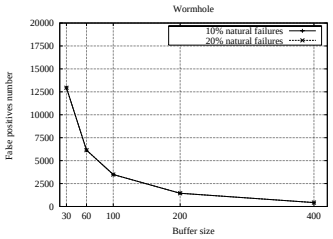

Figure 8: Number of false positives of wormhole attack

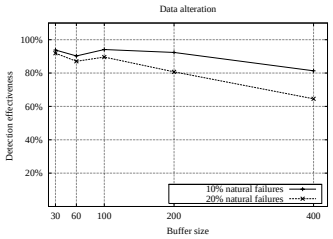

Figure 9: Detection effectiveness of data alteration attack

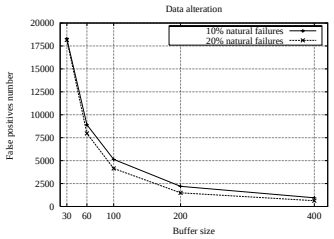

Figure 10: Number of false positives of data alteration attack

6.2.2 Data Alteration

The effectiveness and the number of false positives for this attack are depicted in Figures 9 and 10, respectively. The data alteration attack is confused with data alteration occasional failures. Monitor M_1 presented good effectiveness and precision, identifying the correct attack and the correct intruder node. Monitor M_2 did not identify anything wrong on the network, for reasons already mentioned. There can be observed a trade off between detection effectiveness and the number of false positives.

On this type of attack monitors with smaller buffer sizes have a more accurate detection level. This happens because on larger buffers the generated failures of an attacker do not take the averages of occasional network failures too high, since many messages are used in their computation. However, smaller buffer sizes make the IDS to generate more false positives, because inaccurate averages happen more often. Buffer sizes of 30 and 60 always produce between 5000 and 19000 false positives, which is a number that makes the unreliable IDS.

Detection level is higher when occasional network failures rate is lower and vice-versa. With 20% of failures the buffer sizes of 200 and 400 begin to show a significant detection decrease, with the latter having less than 70% detection. As with the false positives, they happen less often with less frequent occasional failures and smaller buffers. In larger buffers they are always lower.

6.2.3 Blackhole and Selective Forwarding

The effectiveness and the number of false positives for these attacks are depicted in Figures 11 to 14, respectively. These two types of attacks can be confused with the message loss occasional failure. As with data alteration, the attacks were detected correctly by monitor M_1, while monitor M_2 did not notice anything wrong on the network. The detection levels were also close to 100%, with the false positives numbers close to the ones of the data alteration attack type. The larger buffer sizes, however, produced little more false positives than with that kind of attack. The difference from that attack is that the averages of number of messages lost per number of messages received generally are high numbers which vary more. This way, the detection of those types of attacks is more "sensitive", generating more attack reports. With that, an attack almost always increases the failure average on an IDS buffer enough to generate a report. The detection levels are directly proportional to buffer sizes, and this can be seen more clear on simulations with more frequent occasional network failures.

6.2.4 Jamming

The effectiveness and the number of false positives for this attack are depicted in Figures 15 and 16, respectively. The jamming attack can be confused with the message collision occasional failure. It is one of the attacks with better detection results, i.e., with 100% detection in almost every simulation run. The number of false positives is low, similar to the results obtained from the data alteration attack simulation. Like the attacks confused with message loss, detection levels were proportional to buffer levels. As detection levels are almost 100% in any configuration, the only result that changes when the occasional failures frequency decrease is the number of false positives, which also decreases. The detection of jamming attacks is very similar to the detection of attacks that are confused with message loss failures. The difference is that jamming is easier for the IDS to detect, since it only needs to detect collision in one message to report a failure, with no need to compare subsequent messages. Because of that, detection effectiveness is low dependent on buffer sizes.

Besides the correct detection of jamming attacks, other monitors have detected false positives, reporting attacks from innocent nodes such as blackhole and negligence.

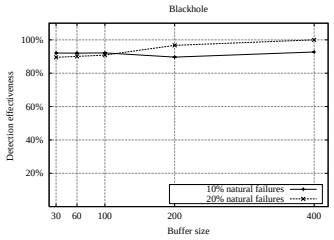

Figure 11: Detection effectiveness of blackhole attack

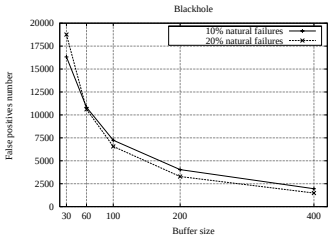

Figure 12: Number of false positives of blackhole attack

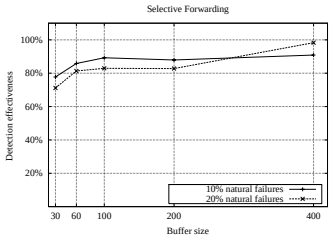

Figure 13: Detection effectiveness of selective forwarding attack

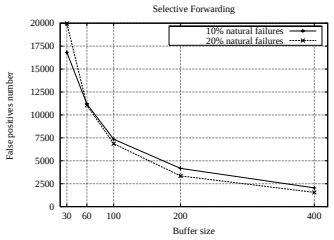

Figure 14: Number of false positives of selective forwarding attack

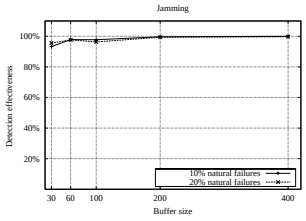

Figure 15: Detection effectiveness of jamming attack

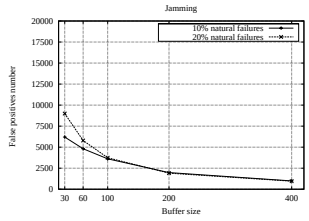

Figure 16: Number of false positives of jamming attack

This happens because accused nodes cannot send their own messages neither retransmit received messages, due to the jamming attack.

Monitors M_1 and M_2 could detect the attack, but they are not able to identify the intruder. However, by observing the innocent nodes accused of blackhole and negligence, one can see that those nodes are exactly the neighbors of the real intruder. In this way, at least the area of the attack can be determined.

6.3 Energy Consumption

In our experiments, energy consumption was considered in the following situations: message transmission, message receiving, and message listening. While the first two are common tasks to a sensor node, the latter activity is described as the process of verification of the beginning of a message header being received, followed by its discard when it is discovered that the message is not addressed to the receiving node or does not concern it in case of forwarding to the base station. By doing this, energy can be saved and the network lifespan is increased.

Messages of 36 bytes[1] were considered, in which 2 bytes were used as the immediate destination address (next hop). As defined in [23], the data transmission rate is 0.26 $\mu s/bit$. Considering the electrical current that flows through each node while receiving (7.0 mA) and sending (21.5 mA on the highest power) messages, also obtained through experiments in [23], the energy consumption is defined as follows:

- $Q_{Transmission} = 3 \times 21.5mA \times (0.26 \times 10^{-6}s/bit \times 288bits) = 0.48375\ mJ/message$;
- $Q_{Reception} = 3 \times 7.0mA \times (0.26 \times 10^{-6}s/bit \times 288bits) = 0.1575\ mJ/message$;
- $Q_{Listening} = 3 \times 7.0mA \times (0.26 \times 10^{-6}s/bit \times 16bits) = 0.00875\ mJ/message$;

[1]the same size used in several TinyOS applications [1]

where dissipated energy (Q) = Power × Electrical Current × Time, and Time = Transmission Rate × Message Size.

In order to measure the energy consumption of our IDS system, two scenarios were proposed. In the first scenario, there were no monitor nodes in the network, and the consumption presented was calculated from normal network utilization and workload without intruder attacks. In the second scenario, some of the sensor nodes chosen for assessment were replaced by monitor nodes, which are capable of activating their IDSs when necessary, while maintaining the same network topology and node positions. Due to space restrictions, detailed information about these scenarios is not provided in this work. In the first scenario, monitor nodes consume more energy than its common counterparts. When its IDS is functioning, a monitor node listens to and completely receives all messages coming from its neighborhood, even if they are not addressed to it, so it can save them for future detection analysis, which explains its higher energy expenditure. Nodes with higher consumption are those closer to the base station (root of the routing tree).

In both scenarios, energy consumption is deeply connected to the placement of sensor nodes into the routing tree. However, it can vary according to network topology and protocol utilized for communication. As said before, a higher energy consumption is related to the promiscuous listening mode of monitor nodes. It is important to point out that we have not considered the energy consumption for activities related to message processing by the IDS and sensor nodes, which could give us a better approximation to a real network, and will be treated in the future work.

7. CONCLUSION

The goal of this work was the study of intrusion detection in WSNs and the associated design of an IDS considering the restrictions of such networks. The proposed IDS is "based on the specification" [11, 3, 24], since the WSN may vary depending on the application goal. We have outlined a method for generating specific IDSs based on the target WSN that can become automatic in the future.

Our detection is decentralized since the IDSs are distributed on network, installed in common nodes. The collected information and its treatment is performed in a distributed way. Distributed Systems are more scalable and robust since they have different views of the network. Besides, the IDS can notice the attack fast because the monitor is near to the intruder (their distance is one hop, since the monitors were distributed in order to cover all network nodes).

8. REFERENCES

[1] *http://www.tinyos.net/.*

[2] Y. an Huang and W. Lee, *A cooperative intrusion detection system for ad hoc networks*, in Proc of the 1st ACM Workshop on Security of Ad hoc and Sensor Networks, 2003, pp. 135–147.

[3] I. Balepin, S. Maltsev, J. Rowe, and K. Levitt, *Using specification-based intrusion detection for automated response*, in Proc of the 6th Int'l Symp on Recent Advances in Intrusion Detection, September 2003.

[4] J. Deng, R. Han, and S. Mishra, *A performance evaluation of intrusion-tolerant routing in wireless sensor networks*, in Proc of IEEE 2nd Int'l Workshop on Info Processing in Sensor Networks, April 2003, pp. 349–364.

[5] Y.-C. Hu, A. Perrig, and D. B. Johnson, *Packet leashes: A defense against wormhole attacks in wireless networks*, in Proc of IEEE Infocomm 2003, 2003.

[6] M.-Y. Huang, R. J. Jasper, and T. M. Wicks, *A large scale distributed intrusion detection framework based on attack strategy analysis*, Computer Networks, 31 (1999), pp. 2465–2475.

[7] K. Ilgun, *Ustat: A real-time intrusion detection system for unix*, in Proc of IEEE Computer Society Symp on Research in Security and Privacity, May 1993.

[8] K. Ilgun, R. A. Kemmerer, and P. Porras, *State transition analysis: A rule-based intrusion detection approach*, IEEE Trans on Software Engineering, 21 (1995), pp. 181–199.

[9] C. Karlof, N. Sastry, and D. Wagner, *Tinysec: A link layer security architecture for wireless sensor networks*, in Proc of the 2nd Int'l Conf on Embedded Networked Sensor Systems, 2004, pp. 162–175.

[10] C. Karlof and D. Wagner, *Secure routing in wireless sensor networks: Attacks and countermeasures*, in Proc of 1st IEEE Int'l Workshop on Sensor Network Protocols and Applications, 2003.

[11] C. Ko, M. Ruschitzka, and K. Levitt, *Execution monitoring of security-critical programs in distributed systems: a specification-based approach*, in Proc of the 1997 IEEE Symp on Security and Privacy, 1997, p. 175.

[12] S. Kumar and E. H. Spafford, *A software architecture to support misuse intrusion detection*, in Proc of the 18th Nat'l Info Security Conf, 1995, pp. 194–204.

[13] P. Levis, N. Lee, M. Welsh, and D. Culler, *Tossim: Accurate and scalable simulation of entire tinyos applications*, in Proc of the 1st Int'l Conf on Embedded Networked Sensor Systems, 2003, pp. 126–137.

[14] S. Marti, T. J. Giuli, K. Lai, and M. Baker, *Mitigating routing misbehavior in mobile ad hoc networks*, in Mobile Computing and Networking, 2000, pp. 255–265.

[15] M. H. T. Martins, A. P. R. da Silva, A. A. F. Loureiro, and L. B. Ruiz, *An IDS simulator for wireless sensor networks*. Sensornet Technical Report, Comp Sci Dept, Federal University of Minas Gerais, May 2005.

[16] S. Park, A. Savvides, and M. B. Srivastava, *Sensorsim: A simulation framework for sensor networks*, in Proc of the 3rd ACM Int'l Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2000, pp. 104–111.

[17] V. Paxon, *Bro: A system for detecting network intruders in real-time*, in Proc of USENIX, USENIX - Security, 1998.

[18] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, *Spins: Security protocols for sensor networks*, Wireless Network Journal, 8 (2002), pp. 521–534.

[19] W. R. Pires, T. H. P. Figueiredo, H. C. Wong, and A. A. F. Loureiro, *Malicious node detection in wireless sensor networks*, in 18th Int'l Parallel and Distributed Processing Symp, 2004.

[20] P. A. Porras and P. G. Neumann, *Emerald: Event monitoring enabling responses to anomalous live disturbances*, in Proc of 20th NIST-NCSC Nat'l Info Systems Security Conf, 1997, pp. 353–365.

[21] L. B. Ruiz, J. M. S. Nogueira, and A. A. Loureiro, *Manna: A management architecture for wireless sensor networks*, IEEE Comm Magazine, 41 (2003), pp. 116–125.

[22] A. Segall, *Distributed network protocols*, IEEE Trans on Info Theory, 29 (1983), pp. 23–35.

[23] V. Shnayder, M. Hempstead, B. rong Chen, G. W. Allen, and M. Welsh, *Simulating the power consumption of large-scale sensor network applications*, in Proc of the 2nd Int'l Conf on Embedded Networked Sensor Systems, 2004, pp. 188–200.

[24] C.-Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt, *A specification-based intrusion detection system for aodv*, in Proc of the 1st ACM Workshop on Security of Ad hoc and Sensor Networks, 2003, pp. 125–134.

[25] A. D. Wood and J. A. Stankovic, *Denial of service in sensor networks*, IEEE Computer, 35 (2002), pp. 54–62.