Lightweight Energy Consumption Based Intrusion Detection System for Wireless Sensor Networks

Michael Riecker
Secure Mobile Networking Lab
TU Darmstadt
michael.riecker@cased.de

Sebastian Biedermann
Security Engineering Group
TU Darmstadt
biedermann@cased.de

Matthias Hollick
Secure Mobile Networking Lab
TU Darmstadt
matthias.hollick@cased.de

ABSTRACT

Wireless sensor networks are increasingly used in industrial settings and in safety-critical applications, generating a financial and social impact. Complementing to cryptographic means to protect the communication, it is desirable to monitor the performance of the system and detect attackers during operation. However, existing intrusion detection systems are too resource-demanding. In this paper, we propose a lightweight, energy-efficient system which makes use of mobile agents to detect intrusions based on the energy consumption of the sensor nodes as a metric. A linear regression model is applied to predict the energy consumption. Simulation results indicate that denial-of-service attacks such as flooding can be detected with high accuracy, while keeping the number of false positives very low.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: Security and protection

General Terms

Security

Keywords

Intrusion Detection, Wireless Sensor Networks, Mobile Agents

1. INTRODUCTION

Wireless sensor networks (WSNs) have become an established technology able to support a wide range of applications. For example, WSNs are used in logistics applications, such as monitoring the cooling chain of perishable products. Service providers are able to offer real-time data about the condition of goods while in delivery.[1] WSNs are also employed to monitor critical infrastructures like bridges. BriMon [2] is such a system, designed to observe the health of railway bridges. In these applications, security is an issue that needs to be addressed. Apart from attacks on the integrity and confidentiality of data, we also have to defend against threats to the availability of the WSN, such as denial-of-service attacks. Most of the security solutions proposed rely on cryptography, for instance when securing the routing protocol and providing data confidentiality. Cryptography helps to achieve a baseline security, but we also need to provide operational security. Often an attacker has physical access to a node and can obtain the key material, thereby becoming a legitimate member of the network. As a result, a WSN must be able to detect insider attacks, which is difficult without an intrusion detection system (IDS).

Multiple IDSs for WSNs have been proposed. Existing systems differ not only in architecture, but in applied detection techniques and detectable attacks. Some are not limited to particular malicious behavior, while others can only detect specific attacks like sinkholes or wormholes. An insider attacker detection algorithm using localized information was presented by Liu et al. [9]. It explores the existent spatial correlation among networking behaviors of sensors in close proximity. In their approach, each sensor monitors the behavior of its immediate neighbors, identifies outliers using Mahalanobis distances, and applies a majority vote to create the final list of outlying sensors. Consequently, this approach requires significant computational effort which can be too heavyweight in certain situations.

The IDS developed by Yu and Tsai [17] analyzes a large number of features (e.g. packet collision ratio, routing cost, power consumption rate, and sensor reading report rate) to build a model of normal behavior. This variety of analyzed features is common for many of the proposed IDSs.

Our goal is to find a minimal metric to detect intrusions. The developed intrusion detection system is characterized by a mobile agent, which moves randomly from node to node, carrying the battery status of the nodes. The battery status is used to estimate the expected power consumption based on past observations, with the assistance of a linear regression model. Intuitively, this metric is suited to detect several denial-of-service attacks aimed at consuming resources. In particular, flooding attacks cause drastic changes in the battery status of various nodes. A system using energy as a metric for intrusion detection has naturally to be designed in an energy-efficient manner. Our system takes this into account, as obsolete information is not transmitted, computational complexity is kept to a minimum, and communication between nodes is not increased. The main contributions of this paper are:

[1]Smart-Trace is available at http://smart-trace.com

SAC'13 March 18-22, 2013, Coimbra, Portugal.

- The feasibility of using mobile agents for intrusion detection in WSNs is demonstrated.
- We show that monitoring the energy consumption is sufficient to reliably detect attacks.
- The complexity of our IDS is stripped down to a minimum, meeting WSN requirements.

The remainder of this paper is organized as follows. We first present our system, its requirements and assumptions in Section 2. Then the usage of mobile agents in WSNs is discussed in Section 3. After reporting simulation results in Section 4, we summarize related work of this paper in Section 5. Finally, Section 6 concludes the paper.

2. SYSTEM OVERVIEW

This section presents our system, states the assumptions, and describes the method applied for intrusion detection.

2.1 Mobile Agents used for Intrusion Detection

One possible classification of current intrusion detection systems for WSNs is according to their architectural design (1) decentralized, (2) centralized, and (3) hybrid systems. The majority of IDSs work in a decentralized fashion, where intrusions are detected locally by the sensor nodes. Therefore, the nodes must install an IDS such as *LIDeA* [7], which consumes about 10 KB out of 48 KB ROM provided by a default node platform such as TelosB. In some approaches nodes need to collaborate and overhear each other's communication, spending a significant amount of energy. Another drawback is that decentralized solutions might be unable to detect certain attacks, as a single node has only local knowledge (including its direct neighborhood).

In a centralized system, all information relevant for intrusion detection has to be transferred to a single point, typically the base station. This can create a large communication overhead and is subject to a single point of failure.

Hybrid IDSs are a combination of centralized and decentralized IDSs. However, they are still in their infancy for wireless sensor networks.

In the following, we propose a new IDS architecture based on mobile agents. Our design has features that respect the unique requirements of sensor networks:

- Efficiency: The mobile agent must not be stored permanently by a node. It carries only necessary data and may delete obsolete information.
- Flexibility: Instead of reprogramming all sensor nodes, a mobile agent is easily changed. Multiple agents can be used with different IDS functionality.

2.2 Assumptions and Requirements

We require our IDS to be very lightweight and energy-efficient. Node storage is valuable, and as such we do not want them to install an IDS permanently. Complex computations should be avoided and the communication overhead must be kept at a reasonable level. Instead of analyzing multiple features, we want to focus on a single metric capable of detecting many types of DoS attacks. In this paper, we demonstrate that the energy consumption is such a metric. The agent movement shall be completely random, i.e. we perform a random walk. All nodes should be visited by the mobile agent, therefore we analyze the random walk on different topologies, such as mesh and random (see Section 3.2). We further require the mobile agent itself and the data it carries to be of reasonable size. For this purpose, the number of energy readings needed to decide whether a node is under attack is minimized. Similarly, the data representation has to be chosen appropriately.

We assume an adversary mounting attacks that influence the power consumption, which is supposed to be relatively constant under normal conditions. We also assume that the attacker may compromise a node and alter its readings, but he cannot change those of other nodes without physically capturing them. Deviations from normal power consumption must be strong enough to be detectable. For example, the attacker might launch a flooding attack, causing a large number of additional messages to be transferred. Even though an attacker might try to evade detection by flooding at a very low rate, we argue that in a DoS scenario an aggressive attacker is more realistic.

We further assume that agents are transferred as regular messages and can be executed on the nodes. In order not to create new attack vectors, we assume means to ensure the integrity of mobile agents and the carried data; one possibility is to use cryptographic hashes. Another assumption is that an attacker cannot remove a mobile agent without evidence, i.e. missing agents can be detected by the sensor network for example with the assistance of beacon messages.

In the case of an attack, a warning should be generated and transmitted to the base station. As we focus on intrusion detection, this part is beyond the scope of this paper.

2.3 Energy-based Intrusion Detection

Instead of sending data to other nodes and/or the base station for intrusion analysis, our proposed IDS is based on mobile agents that visit nodes in a random walk and collect their energy statuses. Our system thus exchanges the overhead of installing a local IDS with the overhead of sending/receiving an agent. We then use a linear regression model to predict the normal energy consumption for each node independently, i.e. we take into account that the energy consumption varies across nodes, e.g. nodes near the base station often exhibit higher load. If a node's energy consumption is deviating significantly, i.e. it consumes either too much or too little energy, a warning is generated. We want to emphasize that our system is not limited to the energy as metric for intrusion detection; other metrics such as CPU load might be applied as well.

We argue that some severe attacks can be detected by solely using the energy consumption as a single metric. One such attack is the exhaustion attack, in which an attacker exploits repeated collisions, causing resource exhaustion, by letting a node retransmit the packets continuously [16]. Attacks on the routing protocol are also detectable. In a sinkhole attack, all traffic from a large area flows through the attacker, thereby increasing the energy consumption of nodes on the route. Similarly, in a blackhole attack, the malicious node drops packets, thus hindering the destination from receiving the packets and decreasing its energy consumption. It is possible for jamming attacks to be discovered by the receiving node, since the number of messages received decreases or even drops to 0, causing energy to be saved at that node. Clearly, flooding attacks are well-suited for de-

tection through changes in the energy status. Our approach is nevertheless not supposed to be a general security framework capable of defending against all types of attacks, such as sybil or data manipulation attacks.

Linear Regression.

To determine a range in which the energy consumption of a node should fall we apply a linear regression model. The mobile agent produces a data set denoted

$$D = \{(x_{ij}, y_{ij}) | i = 1, 2, ...; j = 1, 2, ..., M\}$$

Each x_{ij} corresponds to the energy status of node j at the ith visit, and y_{ij} corresponds to the energy status predicted at the $(i-1)$th visit. For each node j, we learn a target function f_j which maps the observations x into the prediction y. As we assume a linear relation between x and y, we can write the general form of the regression equation as

$$f_j(x) = b_j + a_j x = y \quad (1)$$

In Equation 1, the parameter b_j is the y intercept of the linear model, and a_j is the slope. We minimize the sum of the squares of the differences between the predicted and the actual values. At the ith visit, b_j and a_j are calculated using the last K pairs of (x,y). The slope is calculated as

$$a_j = \frac{K \sum_{t=i+1-K}^{i} x_{tj} y_{tj} - \sum_{t=i+1-K}^{i} x_{tj} * \sum_{t=i+1-K}^{i} y_{tj}}{K \sum_{t=i+1-K}^{i} x_{tj}^2 - (\sum_{t=i+1-K}^{i} x_{tj})^2}$$

The intercept is then given by

$$b_j = \frac{\sum_{t=i+1-K}^{i} y_{tj} - \sum_{t=i+1-K}^{i} a_j x_{tj}}{K}$$

This linear regression with ordinary least squares has a time complexity of $O(MK)$. The effect of the size of K (called *history size*) on the detection time is studied in Section 4.

3. PRACTICAL CONSIDERATIONS OF MOBILE AGENT BASED IDS

We now analyze the use of mobile agents in a sensor network in terms of energy consumption and agent movement.

3.1 Energy Consumption

To show that mobile agents can be used in sensor networks, we analyze their energy consumption.

Scenario.

The system presented in this paper is not suited for every WSN scenario. While it works very well in environmental monitoring settings with regular measurement and transmitting patterns as in [15], the opposite is true for example in area surveillance scenarios, where a detected event is resulting in intense activity.

Agent Energy Consumption.

As pointed out by Piotrowski et al. [12], a TelosB node has about 6750 J of usable energy. If we assume the agent to be 1 kb of size and using the power consumption per bit as presented in [12], we conclude that receiving one agent needs 1851 μJ, while sending one agent consumes at most 1712 μJ at highest transmission speed and 0 dBm transmit power. These costs only involve the radio energy.

Implementation Aspects.

The implementation of our IDS on real sensor nodes is ongoing work. Apart from the mobile agent code itself, the energy values are transmitted alongside. In order to distinguish the agent code from regular data, e.g. sensor readings, we use the virtual channel concept of Contiki's Rime communication stack. Packets containing the agent and the collected data are tagged with a special identifier, that allows the receiving node to determine if it received a regular packet or an agent packet. We use a 4 bytes field for the total energy consumption and an index to identify specific readings. The index field also has a length of 4 bytes, leading to a total memory amount of 8 bytes for one energy reading. Integrity mechanisms will add additional overhead, for instance HMAC-SHA1 requires 20 bytes. That means, if the history size is set to 3, the data section of the mobile agent reaches 1008 bytes in a 12-node network.

3.2 Agent Movement

One crucial design decision in a mobile agent based system is the agent movement. We decided to let the agent perform a random walk. In the following, we present basic facts and simulation results about how often a node is visited by the mobile agent in different topologies.

We define $G = (V, E)$ as a connected graph that has n nodes and m edges. If we start a random walk on G at a node v_0, we walk to a neighbor of v_0 with probability $1/d(v_0)$, where $d(v_0)$ is the degree of v_0. Because the transition from one node to another is independent of the preceding and succeeding transitions, we can assume a Markov chain. For this Markov chain, let $M = (p_{ij})_{i,j \in V}$ be the matrix of transition probabilities with

$$p_{ij} = \begin{cases} 1/d(i) & \text{if } i, j \in E \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

From Equation 2 we can observe that every edge is passed through with the same frequency, i.e. $2m$, since $\pi(i)p_{ij} = 1/(2m)$ for $i, j \in E$, given a stationary distribution [10].

In our experiments, we simulated a network with 12 sensor nodes arranged (1) in a mesh and (2) in a random topology, an example is shown in Figure 3 and Figure 1 respectively. The arrows indicate a link between two nodes, i.e., the nodes being in transmission range of each other. From Figure 2 we observe the influence of the node degree; for instance, node 1 is visited approx. twice as often by the mobile agent as node 2 in the random topology. This is due to the higher degree of node 1. Similar results are obtained for the mesh topology, in which a node at the border (like node 8) is not visited as regularly as a node in the center (like node 5).

As a consequence, an attack started by a node with a low degree is likely to take more time to be detected. However, nodes with many links are more attractive to an attacker, as the effects of an attack propagate faster.

4. SIMULATION STUDY

We now describe our simulation-based evaluation. First, we show the feasibility of our approach. Then we describe the determination of optimal values for the critical parameters of our simulation, namely *migration time*, *slope*, and *history size*. Further, we evaluate the proposed detection algorithm with regard to detection accuracy in a scenario with a flooding and a blackhole attack. To conclude, the

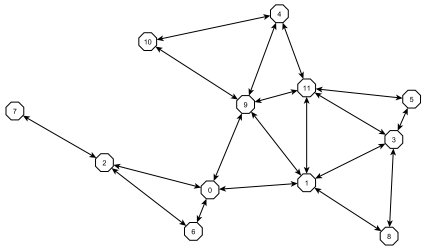

Figure 1: WSN with 12 nodes and random topology.

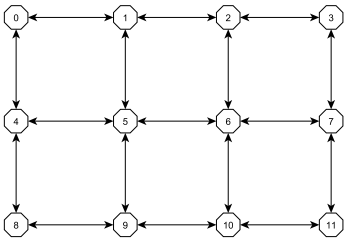

Figure 3: WSN with 12 nodes and mesh topology.

influence of the history size on the detection time is studied. We summarize our findings in Table 1.

For all simulations, we extended an existing WSN simulator, designed for the STEF scheme [6] and developed in Java, to be used with mobile agents. The simulation platform is a standard ThinkPad W500 laptop with 4 GB RAM and an Intel Core 2 Duo T9600. The simulated sensor nodes correspond to TelosB nodes; their energy consumption rates were obtained from [13]. Currently, we only simulate the energy consumption of the radio chip CC2420 as this is the largest energy driver in a WSN. As routing protocol we implement AODV. The agent has a size of 1 kb.

4.1 Initial Demonstration

Simulations were performed to analyze the influence of a flooding attack on the energy consumption of all nodes in the network. The goal is to detect attacks based on the energy consumption on either the malicious node itself or indirectly on its neighbors. The simulated WSN consists of 12 nodes arranged in a mesh topology (see Figure 3; the arrows indicate a link between two nodes, i.e., the nodes being in transmission range of each other) with each node sending one random message per minute. After 128 minutes, node 3 starts an attack by flooding an additional message with a random destination to the WSN every minute.

Figure 4 shows the energy consumption (averaged over four test-runs) of each node. A very important finding is that this attack does not only influence the energy consumption of the malicious node itself, but it also affects all other nodes. The increase in energy consumption is especially significant for the direct neighbors of the attacking node, namely nodes 2 and 7. Those nodes receive a considerably higher number of messages than before launching the attack. Thus, the performed flooding attack can be detected not only by analyzing the energy slope of the malicious node itself, but also by monitoring other nodes in the WSN which have abnormal energy consumptions. This is important, as the attacker is likely to report fake data to the mobile agent.

4.2 Determination of Parameters

To reliably detect attacks, we need to determine meaningful values for the parameters used in our approach, in particular the *warm-up phase* and the *slope limit*, which should not be exceeded without triggering an alert.

Before calculating the energy slopes, we need a warm-up phase expressed as the number of agent migrations, in which the mobile agent collects multiple energy values from each visited node. The warm-up phase should guarantee that the agent collects as many energy readings of each node as required by the history size. Because the agent moves randomly, the number of migrations has to be higher than the number of nodes multiplied by the size of the history. Additionally, as shown in Section 3.2, the visit frequency depends on the node degree. To account for this, we need another multiplier k. In our simulations, $k = 2$ led to a high probability of collecting sufficient energy values from all nodes, and is therefore used in the rest of this paper. Thus, we calculate the number of agent migrations needed before starting the detection mechanism following Equation 3.

$$warmup_{migrations} = number_{nodes} \times size_{history} \times k \quad (3)$$

Next, we have to determine a value for the slope limit, which is divided into *upper* and *lower* slope limit to take into account attacks causing a lower energy consumption by e.g. discarding packets. Exceeding or falling below those limits is regarded an anomaly. For this purpose, we performed additional simulations. The simulated sensor network consists

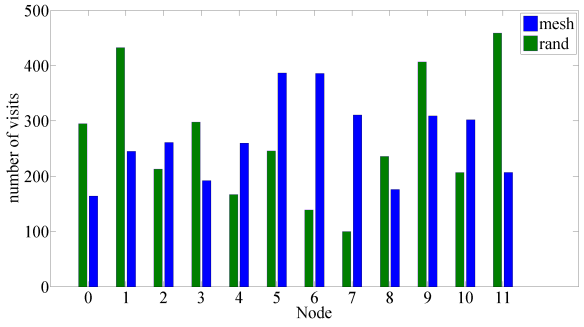

Figure 2: Average distribution of visits of a random walking agent.

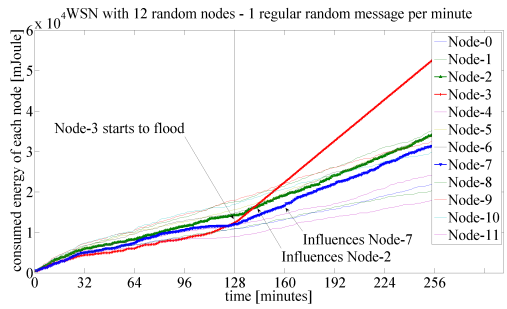

Figure 4: Measured energy of a WSN with 12 nodes arranged in a mesh.

Table 1: A summary of the major simulation results of this paper

Objective	Section	Conclusion
Feasibility	4.1	By launching a flooding attack we show that the energy consumption is a feasible metric for detecting intrusions.
Parameters	4.2	We determine optimal parameters for our simulated network. They are unique for each WSN.
Detection Accuracy	4.3	Flooding and blackhole attacks can be detected with a low false-positive rate.
History Size	4.4	Larger history sizes increase the detection time.

of 12 randomly placed nodes. Each node injects one random message per minute into the network. Every minute, the agent migrates randomly from node to node. The history size is set to three, which means the agent carries the measured energy values of the last three visits of each node.

Figure 5 shows the current slopes of the collected energy values calculated by our randomly migrating agent (averaged over four test-runs). Immediately after the start, the WSN is not balanced and there are relatively high slopes in the energy consumption due to the nature of the managing phase of the underlying routing protocol. Since we use AODV, the nodes need to establish routes and create routing tables after the start-up, resulting in a message overhead.

After the warm-up phase, which lasts for 72 migrations following Equation 3, the WSN becomes more balanced and the average slope of all nodes is stabilizing. Considering the results of Figure 5, we choose a value of 0.3 for the upper slope limit, i.e. the limit that applies to attacks increasing the energy consumption. Furthermore, we use a lower slope limit of 0.02, i.e. the minimal slope each node should have; otherwise an attack might be underway. Such an attack resulting in an abnormal low energy consumption could be a blackhole attack (see Section 4.3).

4.3 Detection Accuracy

To evaluate the effectiveness of our approach, we measure the detection accuracy when performing a flooding and a blackhole attack. The false-positive rate is therefore tracked and we determine the average number of migrations until detection. As a false-positive in the flooding attack scenario, we count all cases in which a node that is not the attacking node itself or a direct neighbor is reported as anomalous. Therefore the false-positive rate is rather pessimistic, as effects of the attack might also propagate to 2-hop neighbors. Regarding the blackhole attack, a false-positive is reported whenever a node other than the attacker is wrongly identified as suspicious. Throughout all simulations in both attacking scenarios we were always able to detect the attack, i.e. the true-positive rate is 100%.

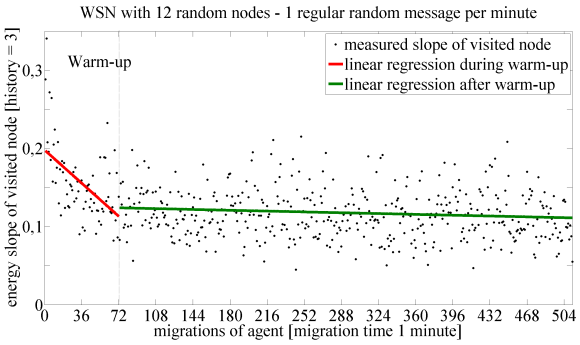

Figure 5: Energy slopes in a WSN with 12 nodes and random topology.

In our simulations, we vary the number of nodes from 12 to 36 and the density of the network from sparse to dense in order to demonstrate the scalability of our approach. Another parameter we vary is the migration time of the agent.

Flooding Attack.

We performed a flooding attack in WSNs with 12, 24, and 36 nodes arranged in a random topology; an example topology is shown in Figure 1 in Section 3.2. Each node is transmitting one message per minute. The parameters used are listed in Table 2.

Table 2: Values used for attack simulation

No. of nodes	Warm-up	History size	Slope limit
12	72 migrations	3	0.3
24	144 migrations	3	0.3
36	216 migrations	3	0.3

Immediately after the warm-up phase, a randomly selected node starts a flooding attack by sending an additional random message every minute. Our agent uses different migration times of 30s, 60s, 90s and 120s. Moreover, the false-positive rate is tracked. Figure 6 shows the averaged results of the simulations (for each network size and for each migration time we performed 32 different test-runs). The false-positive rate can be observed in Table 3.

A main finding is that having a high migration rate leads to a higher false-positive rate. This is due to the fact that the mobile agent needs comparatively more energy for its own operation, thereby interfering with the normal energy consumption of the wireless sensor network. To compensate for this, the agent could subtract its own energy consumption from the nodes' energy readings. However, the agent would have to keep track of the number of visits of each node, as the nodes themselves do not consider the agent's energy consumption. In our ongoing implementation on real nodes, we follow this approach and let the agent handle its energy consumption internally.

In contrast, using a lower migration rate (lower than the

Table 3: False-positive rate for the flooding attack

No. of nodes	30s	60s	90s	120s
12	46.9%	37.5%	12.5%	6.3%
24	46.9%	31.3%	6.3%	3.1%
36	53.1%	40.6%	9.4%	3.1%

normal message rate of the nodes) leads to a very low false-positive rate and a useful detection rate of the flooding attack. For instance, with a migration time of 120s, it takes 24 migrations in a 12-node network until detection while at the same time the false-positive rate reaches 6.3%. In our setting, a migration time of 120s led to the best results with regard to the false-positive rate. If we accept a slightly higher false-positive rate, the detection time can be reduced by changing the migration time to 90s. The number of migrations needed to detect the flooding attack increases linearly with the number of nodes, thus our approach is scalable. The 95% confidence intervals show that the variation of the average migrations until detection increases with the network size: depending on the location of the attacker (e.g. isolated at the corner with very few direct neighbors), the number of migrations until detection can be significantly higher in larger networks.

Another factor that could potentially influence the detection rate is the network density. For a wireless sensor network consisting of 12 nodes arranged in a random topology we varied the density from two direct neighbors on average to three direct neighbors on average. The results are presented in Figure 7. A dense network achieves better detection rates in terms of migrations until detection than a sparse network. For example, a migration time of 120s leads to approximately 46 and 33 migrations until detection in a sparse and dense network, respectively.

Blackhole Attack.

The flooding simulation setup presented above was also used to evaluate the detection of a blackhole attack. Hence, the parameters of Table 2 remain mostly unchanged. The only difference is that the slope limit is now set to 0.02, as the blackhole attack should decrease the energy consumption (of the destination nodes and the nodes not on a path to the attacker) due to dropped packets. Figure 8 shows the results of the simulations. Again, for each network size and for each migration time we performed 32 test-runs. Compared to the results of the flooding attack, the agent needs more migrations until detection. This is because the blackhole attack does not influence the energy consumption of the attacker's neighbouring nodes as significantly as the flooding attack.

In a blackhole attack, the regular nodes have to be inactive, i.e. not receiving packets, for a certain period of time until the slope drops. This epoch is fixed and therefore a higher migration speed cannot compensate the effect

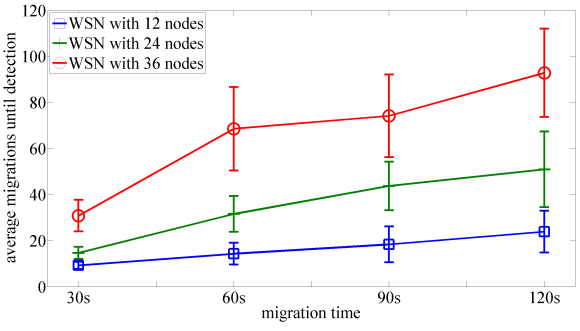

Figure 6: Average migrations until detection of a flooding attack.

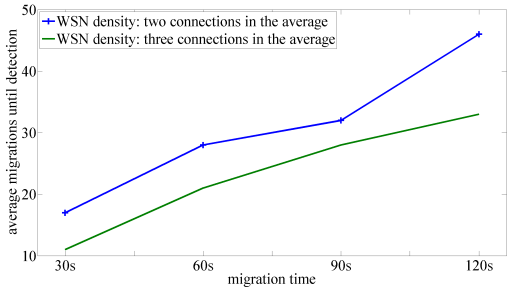

Figure 7: Average migrations until detection of a flooding attack in a WSN with 12 nodes and a random topology. Different node density.

Table 4: False-positive rate for the blackhole attack

No. of nodes	30s	60s	90s	120s
12	37.5%	15.6%	9.4%	3.1%
24	34.4%	21.9%	6.3%	3.1%
36	40.6%	15.6%	6.3%	3.1%

of the increasing number of migrations until detection. Instead, the slower the agent migrates, the less migrations (and therefore less energy) are needed for detection.

Regarding the false-positive rate (Table 4), we achieve slightly better results than with the flooding attack. The lowest false-positive rate is 3.1% for a migration time of 120s throughout all network sizes.

4.4 Influence of the History Size

One way to optimize our IDS and keep it as lightweight as possible, is to change the history size, i.e. the number of readings needed from each node to predict the energy consumption. In simulations for a 12-node WSN we varied the history size from 3 to 4 and 5, while performing a flooding attack. Figure 9 presents the results showing that the average number of migrations until detection increases with the history size. A smaller history size is able to reflect the change in the energy consumption more quickly, thereby generating an anomaly alert faster than a larger history size.

5. RELATED WORK

Our work covers several aspects: first the mobile agent paradigm, second the use of mobile agents to detect intru-

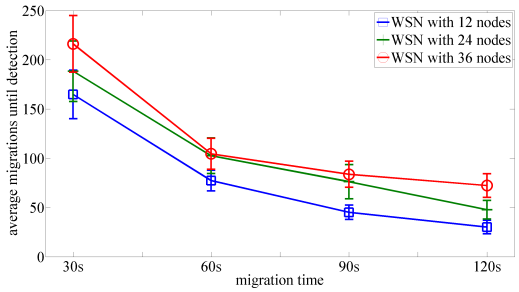

Figure 8: Average migrations until detection of a blackhole attack.

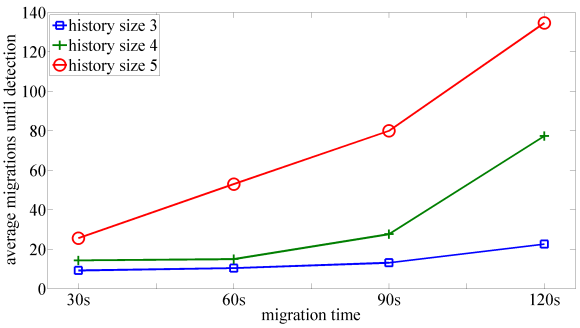

Figure 9: Influence of the history size. WSN with 12 nodes in a random topology and a flooding attack.

sions, and third the energy consumption as a possible metric for attacks. The combination of these techniques is unique in wireless sensor networks. In this section we discuss related work for these aspects.

Mobile Agents in WSNs.

With *Agilla* a middleware for wireless sensor networks was presented by Fok et al. [3] which allows the development and usage of mobile agents. Instead of deploying pre-installed applications, mobile agents perform the desired tasks. They model each agent as an autonomous entity, but provide inter-agent communication.

The mobile agent based computing model has been evaluated in the context of collaborative processing in WSNs in the work of Qi et al. [14]. The authors use the execution time and the energy consumption as a metric to evaluate the performance of the client/server-based and mobile-agent-based models. In their simulations they found that even though both the execution time and the energy consumption grow as the number of nodes increases, the growth is much faster for the client/server model. They conclude that the mobile-agent-based model does not always perform better than the client/server-based model, but it may be advantageous depending on the use case.

Intrusion Detection using Mobile Agents.

Applying mobile agents for intrusion detection has been widely researched; several works exist. Helmer et al. [4] designed an intrusion detection system for traditional networks employing static as well as mobile agents. Stationary agents reside at each monitored component, gathering information e.g. from system logs and providing this information in a common format. Mobile agents travel between monitored components, classify the data collected from the static agents as normal data and data signifying an intrusion, and pass this information to so-called *mediators*. Mediators manage the mobile agents and further use data mining techniques to relate single events to a specific attack.

Kachirski and Guha [5] proposed a mobile agent based intrusion detection system for wireless ad hoc networks. They use different sensor types to perform specific functions. While few nodes have agents for network packet monitoring, every node's agent monitors the host itself for suspicious activities such as unusual user operations (e.g. invalid login attempts). On a host-level basis, decisions on the threat level of an intrusion are made individually. Certain nodes collaborate in order to make decisions about intrusions affecting the network level. For the purpose of responding and resolving an intrusion, each node is equipped with an action module.

Sparta is a system able to detect intrusions and security policy violations in a network [8]. A pattern language is introduced allowing the user to define intrusion patterns in a declarative manner. The approach to spot these patterns is fully distributed, utilizing mobile agents to correlate event data gathered on the single hosts.

All three approaches mentioned above were not designed for WSNs and create significant overhead.

Energy Consumption.

Some authors have already used energy consumption as a metric for intrusion detection. Nash et al. [11] presented an IDS for mobile computers that uses several parameters like CPU load and disk read and write access to estimate the power consumption. The linear regression they used is therefore quite complex, while our model only needs one variable: the energy status of a node. Moreover, their system determines the energy consumption on a per process basis. This renders the IDS vulnerable to attacks that distribute the workload to many different processes.

Buennemeyer et al. [1] developed an IDS that creates a power profile for mobile devices, generating an alert in case of abnormal current changes. The threshold value is adapted dynamically to account for false-positives and false-negatives. Battery readings are transmitted to a central point, which would cause significant overhead in WSNs. In order to create a reliable profile, a large amount of data has to be sent and analyzed. At the central server, attack traffic is correlated with Snort alerts. Thus, their system is very complex as it combines anomaly detection with a rule-based IDS. Another drawback of the last two approaches lies in the necessity to run an IDS agent on every node. In contrast, our system takes into account specific WSN requirements (typically, nodes run a single application, have severe resource restrictions, sleep most of the time etc.) by developing a lightweight method for intrusion detection.

6. CONCLUSION

In this paper we propose a novel lightweight IDS for wireless sensor networks. We neither require nodes to monitor their environment and collaborate with each other, nor do we need to transfer audit data to a central point. Instead, we use a mobile agent that collects energy readings and raises an alert if sudden changes occur. The feasibility of mobile agents used for intrusion detection in wireless sensor networks has been demonstrated. We further showed that the energy consumption is a suitable metric to detect denial-of-service attacks. In simulations we evaluated our method for intrusion detection and were able to achieve a high detection accuracy while maintaining a low false-positive rate.

7. ACKNOWLEDGMENTS

The work presented in this paper was performed in the context of the Software-Cluster project EMERGENT (www.software-cluster.org). It was partially funded by the German Federal Ministry of Education and Research (BMBF) under grant no. "01IC10S01" and was supported by LOEWE CASED (www.cased.de) and EC-SPRIDE. The authors assume responsibility for the content.

8. REFERENCES

[1] T. K. Buennemeyer, T. M. Nelson, L. M. Clagett, J. P. Dunning, R. C. Marchany, and J. G. Tront. Mobile device profiling and intrusion detection using smart batteries. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, 2008.

[2] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar. Brimon: a sensor network system for railway bridge monitoring. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, 2008.

[3] C.-L. Fok, G.-C. Roman, and C. Lu. Rapid development and flexible deployment of adaptive wireless sensor network applications. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005.

[4] G. Helmer, J. S. K. Wong, V. Honavar, L. Miller,Y. Wang. Lightweight agents for intrusion detection. *Journal of Systems and Software*, 67:109–122, 2003.

[5] O. Kachirski and R. Guha. Intrusion detection using mobile agents in wireless ad hoc networks. *IEEE Workshop on Knowledge Media Networking*, pages 153–158, 2002.

[6] C. Krauß, M. Schneider, K. Bayarou, and C. Eckert. Stef: A secure ticket-based en-route filtering scheme for wireless sensor networks. In *The Second International Conference on Availability, Reliability and Security*, 2007.

[7] I. Krontiris, T. Giannetsos, and T. Dimitriou. Lidea: A distributed lightweight intrusion detection architecture for sensor networks. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks*, 2008.

[8] C. Krügel and T. Toth. Sparta - a mobile agent based intrusion detection system. In *IFIP Conference on Network Security*, 2001.

[9] F. Liu, X. Cheng, and D. Chen. Insider attacker detection in wireless sensor networks. In *Proceedings of the 26th IEEE International Conference on Computer Communications*, pages 1937–1945, 2007.

[10] L. Lovász. Random walks on graphs: a survey. In *Combinatorics, Paul Erdös is Eighty*. Janos Bolyai Mathematical Society, 1996.

[11] D. C. Nash, T. L. Martin, D. S. Ha, and M. S. Hsiao. Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices. In *Proceedings of the third IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 141–145, 2005.

[12] K. Piotrowski, P. Langendoerfer, and S. Peter. How public key cryptography influences wireless sensor node lifetime. In *Proceedings of the fourth ACM Workshop on Security of Ad hoc and Sensor Networks*, 2006.

[13] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, 2005.

[14] H. Qi, Y. Xu, and X. Wang. Mobile-agent-based collaborative signal and information processing in sensor networks. *Proceedings of the IEEE*, 91:1172–1183, 2003.

[15] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 51–63, 2005.

[16] Y. Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 8(2), 2006.

[17] Z. Yu and J. J. Tsai. A framework of machine learning based intrusion detection for wireless sensor networks. In *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pages 272–279, 2008.