FI MU

Faculty of Informatics
Masaryk University Brno

Neighbor-Based Intrusion Detection for Wireless Sensor Networks

by

Andriy Stetsko
Lukáš Folkman
Václav Matyáš

FI MU Report Series **FIMU-RS-2010-04**

 May 2010

Publications in the FI MU Report Series are in general accessible via WWW:

`http://www.fi.muni.cz/reports/`

Further information can be obtained by contacting:

**Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno
Czech Republic**

Neighbor-Based Intrusion Detection for Wireless Sensor Networks

Andriy Stetsko, Lukáš Folkman, Václav Matyáš
Faculty of Informatics, Masaryk University
Czech Republic
xstetsko@fi.muni.cz, xfolkman@mail.muni.cz, matyas@fi.muni.cz

May 3, 2010

Abstract

The neighbor-based detection technique explores the principle that sensor nodes situated spatially close to each other tend to have similar behavior. A node is considered malicious if its behavior significantly differs from its neighbors. The detection technique is localized, unsupervised and adapts to changing network dynamics. Although the technique is promising, it has not been deeply researched in the context of wireless sensor networks yet. In this technical report we analyze symptoms of different attacks for the applicability of the neighbor-based technique. The analysis shows that the technique can be used for detection of selective forwarding, jamming and hello flood attacks. We implemented an intrusion detection system which employs the neighbor-based detection technique. The system was designed for and works on the TinyOS operating system running the Collection tree protocol. We evaluated accuracy of the technique in detection of selective forwarding, jamming and hello flood attacks. The results show that the neighbor-based detection technique is highly accurate, especially in the case when collaboration among neighboring nodes is used.

1 Introduction

A wireless sensor network (WSN) is a highly distributed network of resource-constrained and wireless devices called sensor nodes. Each sensor node monitors some physical phenomenons (e.g., humidity, temperature, pressure, light) inside the area of deployment. The collected measurements are sent to a base station. The communication range of sensor nodes is limited to tens of meters and hence not all of them can directly communicate with the base station. Therefore, data are sent hop-by-hop from one sensor node to another until they reach the base station (see Figure 1).

Security becomes an important issue for WSNs and brings new challenges for security engineers. Cryptographic techniques can be used to prevent an external attacker from eavesdropping or altering the ongoing communication. The area of deployment is not usually physically protected and an attacker can easily access the area and capture some nodes. Since they are not tamper-resistant, the attacker can modify the software running on the nodes to launch a variety of internal attacks. In this work jamming, hello flood, selective forwarding, sinkhole, sybil and packet alteration attacks are considered. It is assumed that the number of captured nodes is significantly smaller than the total number of sensor nodes in the network. However, the damage caused by internal attacks might be significant.

Intrusion detection systems (IDSs) are proper mechanisms to defend against internal attacks and they are widely used in conventional networks. However, these IDSs cannot be directly applied to WSNs mainly due to severe limitations of sensor nodes on energy, memory and computational power. It is enough to have several probes in conventional networks if they are placed in traffic concentration points. In WSNs some attacks can be observed only by the neighbors of malicious nodes. Hence, it is assumed that each sensor node runs an IDS agent and monitors its neighbors in a promiscuous mode. The collected data, we believe, should be analyzed locally by a sensor node itself or in cooperation with only nodes from the close neighborhood since communication activity is highly energy consuming – one bit transmitted in WSNs consumes about as much power as executing 800-1000 instructions [5].

In this work the neighbor-based anomaly detection technique [12, 11] is considered. The basic idea is that neighboring nodes should be dealing with similar network traffic and provide similar values from their sensors. If a node's behavior significantly differs from its neighbors, the node is considered malicious. The technique is unsupervised (it

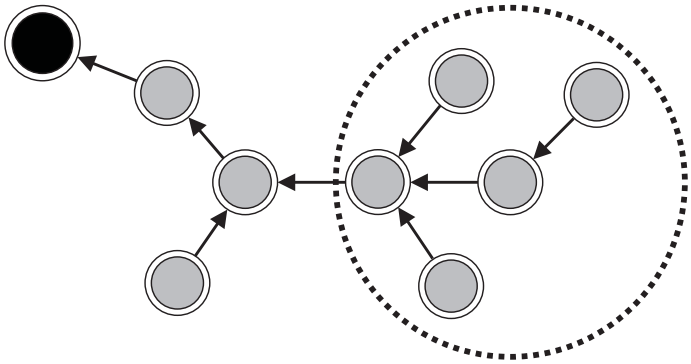


Figure 1: A wireless sensor network with one base station that is depicted as the black filled circle and sensor nodes that are depicted as gray ones. The dotted circle represents the communication range of the sensor node.

does not require prior training), localized and capable of adapting to changing network dynamics. Although these properties are welcome in WSNs, the technique has not been researched deeply yet.

In Section 2 basics of the neighbor-based detection technique and the related work are summarized. Section 3 contains the description of different attacks together with their symptoms and explanation why these symptoms could or could not be used in the neighbor-based technique. In Section 4 we present additional symptoms (and corresponding statistics) which we used in this work for detection of selective forwarding, deceptive, random and reactive jamming attacks. In Section 5 the design and implementation details of the proposed intrusion detection system are provided. We evaluated accuracy of the neighbor-based technique in detection of hello flood, selective forwarding and jamming attacks. The results are summarized in Section 6. We conclude the work in Section 7.

2 Related work

In this section the works of Liu et al. [12] and Li et al. [11] are presented. These works applied the neighbor-based technique to detect insider attacks in WSNs.

2.1 Audit data collection

In [12] every sensor node a_i ($i = \overline{1,n}$) monitors its direct neighbors $N(a_i) = \{b_{i1}, ..., b_{im_i}\}$ in a promiscuous mode. Here n denotes the total number of nodes in a network and

m_i denotes the number of neighbors of the node a_i. Our proposal follows the same approach.

In [11] a network is partitioned into groups. Sensor nodes are in the same group if they are physically close to each other and their sensed data are dissimilar at most δ. The monitoring node a_i collects audit data about its neighbors but only those from the same group.

2.2 Anomaly detection and assumptions

In both [12] and [11] the networking behavior of the node b_{ij} observed by the node a_i is modeled by the q-component vector $f(b_{ij}) = \{f_1(b_{ij}), ..., f_q(b_{ij})\}$ with each component describing the node's activity in one aspect. If the node a_i monitors the nodes $\{b_{i1}, ..., b_{im_i}\}$, it stores the set of the corresponding attribute vectors $F(a_i) = \{f(b_{ij}) : j = \overline{1, m_i}\}$. We use the same notations in this work.

Both schemes [12] and [11] are based on the assumption that in any local area, all attribute vectors $f(b_{ij}) = \{f_1(b_{ij}), ..., f_q(b_{ij})\}$ follow the same multivariate normal distribution.

The node a_i considers the node b_{ij} as suspicious if the (Mahalanobis) distance from $f(b_{ij})$ to the "center" of the set $F(a_i)$ is greater than a predefined threshold. The authors of both works use orthogonalized Gnanadesikan-Kettenring estimation to find the "center" of the set $F(a_i)$.

The authors of [12] claim that nodes can be evaluated in terms of packet dropping rate, packet sending rate, forwarding delay time and sensor readings.

The authors of [11] claim that their scheme can be used to detect fabricated information, jamming, selective forwarding, packet alteration, sinkhole and hello flood attacks by monitoring sensor sensed data, packet sending rate, packet dropping rate, packet mismatch rate, packet receiving rate and received signal strength respectively.

Unfortunately, the authors of [12] and [11] do not provide any notion of circumstances under which the assumption (regarding multivariate normal distribution) holds.

In [12] the experiments were done only on synthetic data which followed multivariate normal distribution. In [11] the experiments were done on real data that contained only sensor readings of humidity, temperature, light and voltage. Such behavioral characteristics as the packet sending rate, packet dropping rate, packet receiving rate and received signal strength remain unexplored.

In this work no assumptions regarding the distribution of attribute vectors are made. A network operating a tree-based routing protocol (without data aggregation) was simulated and the experiment revealed that packet sending rate, packet dropping rate and packet receiving rate did not follow the normal distribution (see Appendix A). Of course, any sample can be approximated by the normal distribution but such approach will result into a high numbers of false negatives and false positives. Unfortunately, Liu et al. [12] and Li et al. [11] did not evaluate their schemes for the packet sending rate, packet dropping rate, packet receiving rate and received signal strength behavioral characteristics.

2.3 Cooperation

In [11] the node a_i alerts other sensor nodes in the same group. Each sensor node in the group decides whether a node is malicious or not based on the number of alerts received from other nodes and its own observations.

In [12] the node a_i broadcasts the list of suspicious nodes within the neighborhood and receives the similar lists from its neighbors. The node a_i considers the node b_{ij} malicious if a majority of sensor nodes $\{b_{i1}, ..., b_{im_i}\}$ suspect the node b_{ij}. In order to increase detection accuracy, sensor nodes share their observed attribute vectors in the close neighborhood. The authors describe how to filter different attribute vectors regarding the same node that is 2-hops away – the vector coming from the most trustworthy relay is chosen.

However, they do not provide any information on how to combine different attribute vectors regarding the same 1-hop neighbor.

In our work several such options are considered (see Section 5.5).

3 Attacks and their symptoms

In this section an overview of attacks together with their symptoms which are commonly used to detect these attacks is provided. The symptoms are expressed by certain behavioral characteristics of a node, e.g., packet sending rate, packet dropping rate, packet receiving rate, received signal strength. The behavioral characteristics that can be used in the neighbor-based detection technique are pointed out.

3.1 Jamming

In a jamming attack a malicious node (or other device) purposefully tries to interfere with physical transmission and reception of wireless communication. The authors of [17] and [16] distinguish four types of jammers. A *constant jammer* continually emits a radio signal or sends out random bits. A *deceptive jammer* constantly injects regular packets to the channel without any gap between subsequent packet transmissions. A *random jammer* alternates between sleeping and jamming randomly. A *reactive jammer* stays quiet when the channel is idle but starts transmitting a radio signal as soon as it senses activity on the channel. During the jamming phase both random and reactive jammers can either behave like the constant or deceptive jammer.

In this technical report the deceptive, random and reactive jammers are considered.

Received signal strength is a measurement of the power present in a received radio signal.

Notation 1. $SS(b_{ij})$ *is the signal strength of the node* b_{ij} *received at the node* a_i.

The distribution of $SS(b_{ij})$ is affected by the constant and deceptive jammers [16]. The node a_i considers the node b_{ij} malicious if the distribution of $SS(b_{ij})$ significantly differs from the distributions $\{SS(b_{i1}), ..., SS(b_{im_i})\}$. The approach cannot be applied to reactive and random jammers. These jammers do not affect the signal strength distribution as they cause channel state to alternate between busy and idle in much the same way as legitimate nodes [16].

Packet sending rate is the number of packets sent over a predefined period of time.

Notation 2. $PS(b_{ij})$ *is the packet sending rate of the node* b_{ij} *observed by the node* a_i.

The deceptive jammer sends an abnormal amount of packets in comparison to legitimate neighboring sensor nodes. The packet sending rate can be used in the neighbor-based detection technique. In order to detect random jammers the number of received packets has to be counted too (see Section 4).

Packet receiving rate is the number of packets received over a predefined period of time. It should be noted that a packet can never reach a destination due to collisions. Hence, a monitoring node might observe retransmissions of the packet. We differentiate two types of the packet receiving rate – with and without retransmissions.

Notation 3. $PR(b_{ij})$ *is the packet receiving rate (the number of retransmissions is included) of the node* b_{ij} *observed by the node* a_i.

Notation 4. $PR'(b_{ij})$ *is the packet receiving rate (the number of retransmissions is not included) of the node* b_{ij} *observed by the node* a_i.

Packet delivery ratio is the ratio of packets that are successfully delivered to their destination compared to the number of packets that were sent by their sender. Packet delivery ratio is calculated as the ratio of the number of received acknowledgements with respect to the number of sent packets.

Notation 5. $PDR(b_{ij})$ *is the packet delivery ratio of the node* b_{ij} *observed by the node* a_i.

The packet acking rate is the number of acknowledgements sent to the node.

Notation 6. $PA(b_{ij})$ *is the packet acking rate of the node* b_{ij} *observed by the node* a_i.

A presence of any type of a jammer in a node's neighborhood can be noticed as the node's packet delivery ratio drops down. However, the attacker itself is not revealed.

Carrier sensing time is the amount of the time spent waiting for gaining the access to the channel. Being under attack of deceptive or constant jammer a legitimate node never finds the medium idle and hence its carrier sensing time is high. This symptom is suitable only for medium access protocols that determine whether the channel is idle by comparing the noise level with a fixed threshold.

Packet send ratio is the ratio of packets that are successfully sent out to the number of packets intended to be sent. If the channel is busy and too many packets are buffered in the medium access layer, newly arrived packets are dropped.

The carrier sensing time and packet send ratio can be monitored only by the particular node itself. Hence, these statistics cannot be used in the neighbor-based detection technique.

3.2 Hello flood

In a hello flood attack [7] a malicious node broadcasts hello packets using a more powerful transceiver than a general sensor node does. Nodes receiving such packets may falsely assume that they are within the radio range of the sender and try to forward their packets through this malicious node. The packets sent will be lost since they will not even reach the malicious node.

Received signal strength measured by the nearest neighbor of a malicious node might be significantly higher than the signal received from other neighbors. The symptom can be used in the neighbor-based detection technique.

3.3 Selective forwarding

The attack is performed by a malicious node that refuses to forward certain packets and simply drops them [7].

Packet dropping rate is the number of packets that were sent to a certain node but were not forwarded by that node.

Notation 7. $PD(b_{ij})$ *is the packet dropping rate of the node* b_{ij} *observed by the node* a_i.

A node that has the packet dropping rate significantly higher than other nodes in the neighborhood is considered malicious. Sensor nodes that are close to each other should have similar packet dropping rates. If a part of a network is congested or affected by environmental changes then majority of nodes in that part will have the packet dropping rates increased.

The drawback of this approach is that an attacker can send a packet to a neighbor that does not exist. If the communication is not encrypted the problem can be solved by broadcasting hello packets in node's 2-hop neighborhood. Having learnt its 2-hop neighbors, an IDS agent can check whether the attacker sends packets to an existing neighbor or not [4]. In our work the attacker is assumed to either drop or send the packet to an existing neighbor.

Packet forwarding rate of a certain node is the number of packets that the node received from the neighbors and consequently forwarded to its parent node during a predefined period of time.

Notation 8. $PF(b_{ij})$ *is the packet forwarding rate of the node* b_{ij} *observed by the node* a_i.

3.4 Sinkhole

In a sinkhole attack a malicious node tries to attract all of the traffic from a particular area towards itself usually by modifying or spoofing routing metrics [7]. An attacker tries to create a sinkhole node from the one that is captured by them. Afterwards, more serious attacks can be run using this node.

If the PR of some node is extremely high, it might be suspected of being the attacker [11]. However, this symptom does not differentiate legitimate sinkholes which may exist depending on a network topology. Furthermore, the PR of surrounding nodes will become high as well because they will gain very good connection to the base station via

the sinkhole node. We do not consider this symptom appropriate for the neighbor-based intrusion detection.

Although the malicious node may claim that its connection to the gateway is better than it actually is, nodes in its neighborhood do not have to change their parents straight away (this depends on a routing protocol, e.g., the CTP for TinyOS is designed this way). The attacker has to downgrade the quality of other nodes' connections in this case. The malicious node may spoof fake root update packets impersonating its neighbors. The authors of [9] suggest that this form of the sinkhole attack may be revealed by an IDS which observes whether senders of root update packets are in the neighborhood of the node running the IDS. If not or if they are even sent by the node running the IDS itself (possible when the packet's header is altered), some node is running the sinkhole attack in the network.

This technique effectively reveals attackers that try to downgrade the quality of other nodes' connections. If some node finds out that another node spoofs packets, it should alert other nodes about this immediately. It does not matter what the common behavior of the neighborhood is. From this point of view, this technique is not suitable for the neighbor-based intrusion detection.

3.5 Sybil

In a sybil attack a malicious node owns several impersonated or fake identities and presents them to other nodes in the network [7]. Sybil nodes can be revealed by position verification techniques. If several nodes are located at the same position, the sybil attack is ongoing with highest probability.

The technique based on measuring the SS is described in [1]. Three cooperating nodes measure the SS upon receiving a message. After exchanging obtained values, the ratio is calculated from them and stored in the message sender's record in a database of neighbors. A unique ratio value determines a unique position of a node.

A similar approach is published in [15] too. The position verification is based on measuring the *time difference of arrival* of packets among cooperating nodes instead of the SS in this case.

The position verification methods need a group of nodes to exchange some information. This information is used for a simple computation whose result is the determination of the position of the node which they exchanged the information about. Each node has a table of such positions of the nodes in its neighborhood. Both techniques do not

look for a property of network behavior which should not vary noticeably for a node in some group of neighboring nodes. Hence, position verification techniques cannot be directly employed in a neighbor-based IDS.

3.6 Packet alteration

In order to detect deliberate alteration of data, an IDS has to store overheard packets in a buffer, wait until the appropriate node forwards them and compare whether the payloads are the same for the forwarded packets and the packets stored in the buffer. The presence of the attacker that alters or spoofs packets is noticed immediately. There is no need to use techniques such as the neighbor-based detection technique to find out if it is common to deliberately alter packets in the node's neighborhood. Deliberate alteration should always be considered as an attack.

4 New Statistics

The simulations revealed that the packet receiving rates of neighboring nodes might differ significantly (see Figure 2). The packet sending, packet dropping and packet forwarding rates of a certain node exhibit the same property since they are implied by its packet receiving rate. Although the packet dropping rate and packet sending rate can be used in the neighbor-based technique for detection of selective forwarding and random jamming attacks respectively (see Sections 3.3 and 3.1), they will result into a high number of false positives and false negatives in networks operating tree-based routing protocols without data aggregation (see Appendix B). In this section we present the statistics which mitigate the mentioned problem to an acceptable level for such networks.

In order to detect the selective forwarding attack we propose to calculate a *packet dropping ratio* – the ratio of the number of dropped packets to the number of received packets without retransmissions. This value should be significantly higher for an attacker when compared to normal nodes from its neighborhood (see Rule 3 in Section 5.3).

In order to detect deceptive and random jamming attacks we propose to calculate a *received-to-sent ratio* – the ratio of the number of received packets to the number of sent packets. The attacker sends some "extra" packets which it did not receive to forward

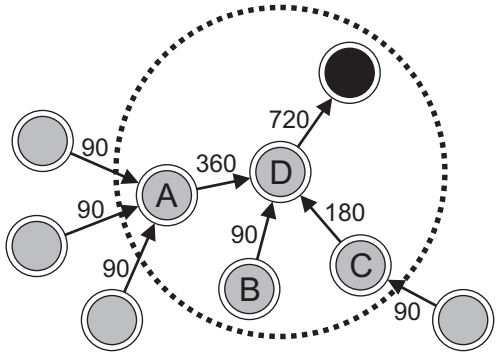


Figure 2: Packet receiving rate. The receiving rates of the nodes A, B and C are 270, 0 and 90 respectively.

and hence it has a lower received-to-sent ratio than it is common in its neighborhood (see Rule 4 in Section 5.3).

In order to detect the reactive jamming attack we propose to use the *packet delivery ratio* (PDR) as it was defined in Section 3.1. The definition of the ratio comforts deployment for detections in CTP operating networks. The PDR should be extremely low for the node that is being jammed (see Rule 5 in Section 5.3).

The packet dropping and packet sending rates of the node b_{ij} observed by the node a_i vary from actual values. Even if the node b_{ij} sends/forwards a packet, the node a_i might not hear that due to different types of collisions which are described in [8]. The packet receiving rate of the node b_{ij} observed by the node a_i varies from actual value too. In addition to collisions, this is also caused by the fact that the node a_i does not hear all the neighbors of b_{ij} (see Section 5.5 for description of nodes' collaboration which is implemented in order to gain the most accurate statistics).

5 IDS design and implementation

The design of our IDS is based on the conceptual architecture proposed in [13]. Every node runs an agent, which monitors the information flowing in its neighborhood. The agent consists of *data acquisition*, *statistics*, *detection*, *alert database* and *collaboration* components. The data acquisition component gathers data from packet headers and stores the processed information in the statistics component. The detection component analyzes the information stored by the statistics component and stores information about suspicious or malicious nodes in the alert database component. If a node must share an event with its neighbors or the base station, it activates the collaboration component.

5.1 Data acquisition

We assume that packet headers are not encrypted and every sensor node monitors its direct neighbors in a promiscuous mode. In order to collect data the code designed and implemented by Honus [6] was used. In the TinyOS, applications do not access a radio hardware directly, but they use virtualized radio abstractions – the `AMSenderC` to send a packet out, the `AMReceiverC` to receive a packet destined to the node and the `AMSnooperC` for snooping on overheard packets that are not destined to this node. The `AMReceiverC` and `AMSnooperC` are directly wired to the `ActiveMessageC` component – the highest layer of the hardware dependent radio stack (see Figure 3). The `AMSenderC` is wired through the `AMQueueC` which provides queueing of outgoing packets. In order to tap the communication the `ActiveMessageC` component is "forged" and put between the original one and virtualized radio abstractions. The `ForgedActiveMessageC` component implements the `AMTap` interface that provides the data acquisition component with possibility to receive packets sent by the node itself and packets overheard in the neighborhood.

The main advantage of the presented approach is that deploying an IDS does not require any changes in the code of the top level applications or in the CTP library.

5.2 Statistics

The statistics component stores the following information about neighboring nodes: active message address of a node, average received signal strength (SS), packet receiving rate (PR), packet sending rate (PS), packet acking rate (PA), packet forwarding rate (PF) and packet dropping rate (PD). All these statistics are collected for a period of predefined duration (denoted as τ) after which detection is started.

In order to keep the track of which packets are being forwarded or dropped the packet store unit was implemented. Packets that were snooped or sent by the node running the IDS are stored in the unit. A packet is erased from the unit if it has been heard to be forwarded. The PF of the corresponding node is increased in such case. The size of the unit is limited to a certain number of packets. If the unit is full and a new packet arrives, the oldest packet is erased and the PD of the corresponding node is increased. Furthermore, if a packet stored in the unit has not been heard to be forwarded within a predefined period of time, it is erased and PD is increased again.

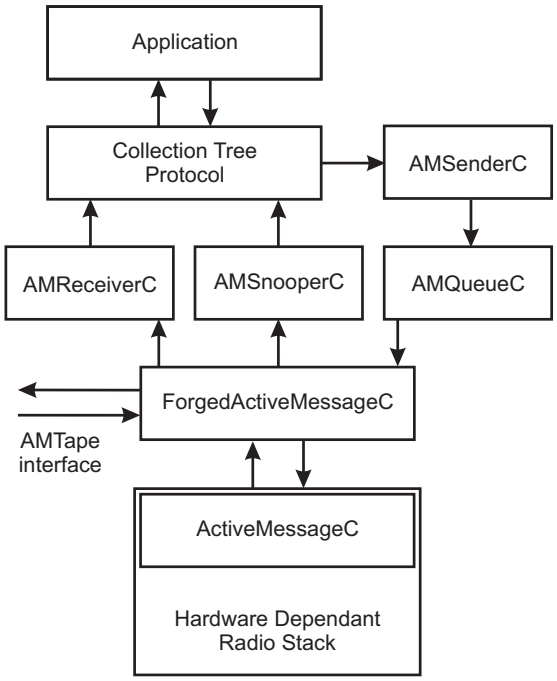


Figure 3: Tapping of the communication. The AMSenderC, AMReceiverC and AMQueueC components should be wired to the "forged" ActiveMessageC component instead of the original one.

5.3 Detection

The detection component is started periodically, after τ has elapsed. Once the information from the statistics component is sorted and stored for the detection, the statistics component's database is cleared and data acquisition for the next detection round can begin. The packets waiting for forwarding in the packet store unit are not touched at all and wait for processing in the next detection round.

The sorting is done according to some attack specific parameter. Then, only the first 20 nodes of the resulting sorted set are passed for the detection itself. The choice of the sorting parameter is done so that the possible attackers are among these 20 nodes with the highest probability. The records about the nodes $\{b_{i1}, ..., b_{im_i}\}$ stored at the node a_i are sorted according to the SS for the detection of the hello flood attack (obviously, the attacker is among the nodes with the highest SS). Nodes having high PR are attractive for the attacker to capture when discussing selective forwarding and so they are sorted according to their PR for the detection of this attack. The deceptive jammer stands out among its neighbors because of its extremely high PS. Even for the case of the random jammer, its PS most probably exceeds the average PS in its neighborhood. Hence, the

statistics are sorted according to the PS in both cases. Sorting according to the PDR is used for the detection of the reactive jamming as the presence of this attack can be noticed based on identifying nodes $\{b_{ij_1}, ..., b_{ij_m}\}$ with extremely low $PDR(b_{ij})$.

Non-malicious nodes are assumed to always prevail in a neighborhood over malicious ones. The node b_{ij} is considered malicious by the node a_i if the Euclidean distance from $f_k(b_{ij})$ to the "center" of the set $\{f_k(b_{i1}), ..., f_k(b_{ij})\}$ is greater than the predefined threshold δ_k. The "center" of the set is defined as the arithmetic average of its elements. Futhermore, from the set of suspicious nodes the nodes having the value of the attribute $f_m(b_{ij})$ lower than γ_m are excluded (especially non-malicious nodes which may cause high occurance of falsely positive alerts). The attribute f_m is chosen per each attack and γ_m has to be set so that only nodes which cause minimal threat are excluded. More formally the detection rule can be written as follows:

$$f_k(b_{ij}) - AVG(f_k(b_{i1}), ..., f_k(b_{im_i})) > \delta_k \ \wedge \ f_m(b_{ij}) > \gamma_m \tag{1}$$

For the detection of the hello flood attack $f_k(b_{ij}) = SS(b_{ij})$ and the rule 1 is as follows (no nodes need to be excluded in this case so γ_m is not introduced):

$$SS(b_{ij}) - AVG(SS(b_{i1}), ..., SS(b_{im_i})) > \delta_{SS} \tag{2}$$

Selective forwarding attack: $f_k(b_{ij}) = \frac{PD(b_{ij})}{PR'(b_{ij})}$ and $f_m(b_{ij}) = PD(b_{ij})$

$$\frac{PD(b_{ij})}{PR'(b_{ij})} - AVG(\frac{PD(b_{i1})}{PR'(b_{i1})}, ..., \frac{PD(b_{im_i})}{PR'(b_{im_i})}) > \delta_{\frac{PD}{PR'}} \ \wedge \ PD(b_{ij}) > \gamma_{PD} \tag{3}$$

Deceptive and random jamming attack: $f_k(b_{ij}) = \frac{PR(b_{ij})}{PS(b_{ij})}$ and $f_m(b_{ij}) = PS(b_{ij})$

$$AVG(\frac{PR(b_{i1})}{PS(b_{i1})}, ..., \frac{PR(b_{im_i})}{PS(b_{im_i})}) - \frac{PR(b_{ij})}{PS(b_{ij})} > \delta_{\frac{PR}{PS}} \ \wedge \ PS(b_{ij}) > \gamma_{PS} \tag{4}$$

Reactive jamming attack: $f_k(b_{ij}) = \frac{PA(b_{ij})}{PS(b_{ij})}$ and $f_m(b_{ij}) = PS(b_{ij})$

$$AVG(\frac{PA(b_{i1})}{PS(b_{i1})}, ..., \frac{PA(b_{im_i})}{PS(b_{im_i})}) - \frac{PA(b_{ij})}{PS(b_{ij})} > \delta_{\frac{PA}{PS}} \ \wedge \ PS(b_{ij}) > \gamma_{PS-RETRY} \tag{5}$$

5.4 Alert database

The alert database component is simply a database where the identifiers of nodes evaluated as malicious are stored. It can be requested for the list of these nodes in order to take some action (e.g., announce them to the base station, exclude them from the routing table). No response actions were implemented in this work.

5.5 Collaboration

Collaboration is used for the information exchange between IDS agents running on different nodes. When network density is low and there is not enough nodes monitored by a single IDS agent, the agent collaborates with its 1-hop neighbors. The neighboring nodes exchange the statistics they have gathered. Alternatively, this can be done among nodes that are two or more hops away from each other. Furthermore, collaboration might be needed even in dense networks where it helps to reveal the most accurate statistics about neighboring nodes. In this case, the IDS agent would not extend the number of nodes it is monitoring but only refine the information about them.

No reputation scheme is implemented in this work and the received information is assumed to be correct (there are no malicious nodes dispatching fallacious information).

The collaboration component starts exchanging messages at the moment when detection is supposed to occur (detection is rescheduled after collaboration has finished). The local statistics are stored in the collaboration database and so the statistics component can be restarted to gather new statistics. Collaboration messages are broadcast to 1-hop neighbors, each describing several monitored nodes. On receiving such a message, the IDS agent stores the records the message is carrying into the collaboration database. The exchange message period ends after a predefined period of time.

It is common that several IDS agents running on neighboring nodes monitor the same node. Several records describing the same node can be found in the collaboration database then. Based on given rules (see below) only one record for each node and each type of an attack is kept. Finally, this is the resulting set passed to the detection component.

The candidate record at the node a_i is chosen from the set of records $\{b_1, ..., b_{m_l}\}$ and is denoted as b in this text. b_m ($m = \overline{1, m_l}$) is the record about the neighbor b_{ij} of a_i received from its neighbor b_{im}. The rules for electing the candidate records are defined in order to comfort detection of specific attacks. The record claiming the highest SS, PR, PS or PA is presumed to be the one closest to the reality. Knowing precise values of these statistics is important for the detection of hello flood, selective forwarding, deceptive or random jamming and reactive jamming attacks respectively (see previous section about statistics sorting done by the detection component for explanation). At the same time, for the case of selective forwarding, deceptive or random jamming and reactive jamming, accurate information about PF, PR and PS respectively is important

too, especially in means of eliminating the number of false positives (see Section 6.1 for the definition of false positives).

The rules are defined to take the records having the values of the first property near to the maximal value of this property. Then, the record having the highest value of the second property is elected from these nodes. The rule differs in the case of reactive jamming. Firstly, the record having the highest PA is chosen and the set of records having their PS close to the PS of this node is computed. The record claiming the highest PDR is chosen among the nodes in this set. The rules are formally stated below. For the case of the hello flood attack it is:

$$SS(b) = MAX(SS(b_1), ..., SS(b_{m_l})) \tag{6}$$

Selective forwarding attack:

$$PR(c_k) > \frac{1}{q} \times MAX(PR(b_1), ..., PR(b_{m_l})) \ \wedge \ PF(b) = MAX(PF(c_1), ..., PF(c_{m_k})) \tag{7}$$

Deceptive and random jamming:

$$PS(c_k) > \frac{1}{q} \times MAX(PS(b_1), ..., PS(b_{m_l})) \ \wedge \ PR(b) = MAX(PR(c_1), ..., PR(c_{m_k})) \tag{8}$$

Reactive jamming:

$$PA(d) = MAX(PA(b_1), ..., PA(b_{m_l})) \ \wedge \ PS(c_k) > \frac{1}{q} \times PS(d) \ \wedge \\ \wedge \ PDR(b) = MAX(PDR(c_1), ..., PDR(c_{m_k})) \tag{9}$$

q is set equal to $1.\overline{33}$ in our implementation.

There is 2-hop neighbors collaboration implemented in our IDS also. It performs in two rounds. The first one is the same as already described in this text. Then, the resulting set of records is said to be the statistics gathered by the actual IDS agent and sent out again in the same fashion as in the first round.

6 Experiments

6.1 Evaluation metrics

In order to evaluate the proposed IDS the number of false positives and the number of false negatives are counted. We defined and discussed these metrics in [14].

If a node considers its neighboring node malicious while it is not, it is considered as a false positive. The number of false positives is the total number of such events in the network. More formal definition follows.

Notation 9. $A = \{a_1, ..., a_n\}$ *is the set of all (monitoring) nodes in the network.*

Notation 10. $B_i = \{b_{i1}, ...b_{im_i}\}$ *is the set of the direct neighbors of the node* a_i *(*$i = \overline{1, n}$*).*

The number of false positives that occurred during the time interval t of duration τ is calculated as:

$FP(t) = \sum_{i=1}^{|A|} \sum_{j=1}^{|B_i|} fp(a_i, b_{ij}, t)$, where

$$fp(a_i, b_{ij}, t) = \begin{cases} 1 & \text{conditions (1), (2) and (3) hold,} \\ 0 & \text{otherwise.} \end{cases}$$

Conditions:

1. The node a_i had an activated IDS agent during the time interval t.
2. The node b_{ij} was not performing an attack during the time interval t.
3. The node a_i made a decision that the node b_{ij} was malicious during the time interval t.

Several false positives may refer to the same node since the node might have more than one neighbor which accuses it falsely. Therefore, additionally the number of different nodes which were accused at least by one of their neighbors is counted.

If a node that runs the IDS agent does not detect its neighboring malicious node, this is considered as a false negative. The number of false negatives is the total number of such events in the network. More formal definition follows.

The number of false negatives that occurred during the time interval t of duration τ is calculated as:

$FN(t) = \sum_{i=1}^{|A|} \sum_{j=1}^{|B_i|} fn(a_i, b_{ij}, t)$, where

$$fn(a_i, b_{ij}, t) = \begin{cases} 1 & \text{conditions (1), (2) and (3) hold,} \\ 0 & \text{otherwise.} \end{cases}$$

Conditions:

1. The node a_i had an activated IDS agent during the time interval t.
2. The malicious node b_{ij} was performing an attack during the time interval t.
3. The node a_i did not make a decision that the node b_{ij} was malicious during the time interval t.

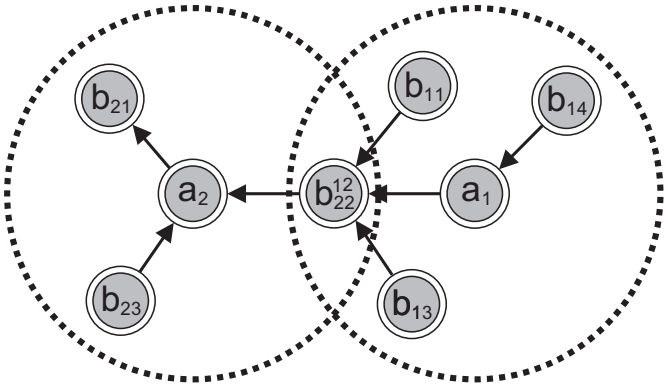


Figure 4: Number of false positives and false negatives.

Several false negatives may refer to the same node since the malicious node might have more than one neighbor which did not detect it. Besides, the number of false negatives per each attacker is counted too and it is compared with the number of IDS agents that successfully revealed the attacker (we talk of the number of correct detections of an attacker in this respect).

For example see Figure 4. The nodes b_{23} and b_{11} are malicious nodes. Other nodes are non-malicious nodes. The node b_{23} was successfully detected by the node a_2, but the node b_{11} was not detected by the node a_1. This results into one false negative and one correct detection in the network. The node b_{22}^{12} was considered malicious by both a_1 and a_2 nodes. This results into two false positives and one falsely accused node.

6.2 Network and IDS configuration

A network with 224 sensor nodes was simulated in the TOSSIM [10] – a discrete event simulator for TinyOS applications. The network was set to low gain mode monitoring [6] with a threshold of -88 dB. The distribution of the number of neighbors in this configuration is depicted in Figure 5. A node had 28 neighbors on average (median is 26 and standard deviation is 13), however, as described in the text above only the subset of the 20 most suspicious nodes from these neighbors was proceeded for detections. Nodes sent the measured values from their sensors to the gateway every 16 seconds (the minimum interval tested by the authors of the CTP [3]).

The IDS did not monitor the network for the very first 304 seconds as that is approximately how long it took the CTP to stabilize the network tree in such network. The detection period τ was set to 304 seconds (hence 19 burst periods were covered by each detection round). The IDS was tested with both shorter and longer detection periods

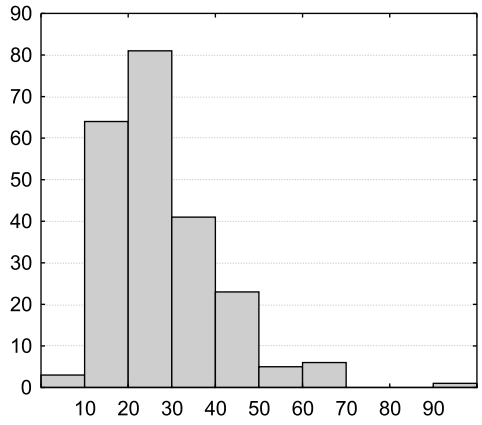


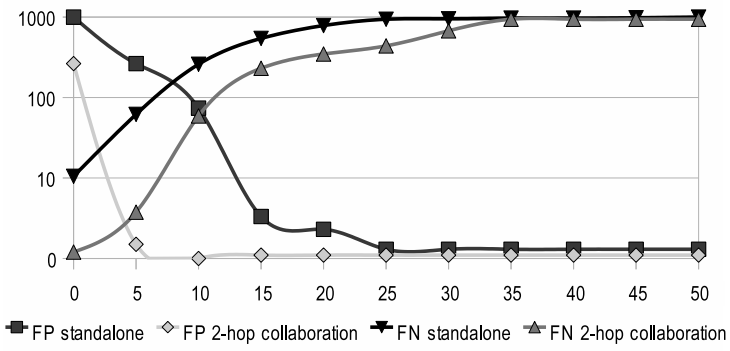
Figure 5: Distribution of the number of neighbors.

(34 and 608 seconds). The 34 seconds long period did not provide good detection results. On the other hand, the 608 seconds long one did not improve results significantly in comparison to the one when τ was set to 304 seconds. Five detection rounds were evaluated in 10 simulations for each attack and results were averaged (they were stable among each detection round and simulation). Both standalone and collaboration (1-hop and 2-hop) detections were simulated using the same data so that results could be compared. Five attackers were involved in each simulation when detections of the hello flood, selective forwarding, deceptive and random jamming attacks were tested. Two nodes were unable to send out packets due to reactive jamming in simulations concerned to this attack.

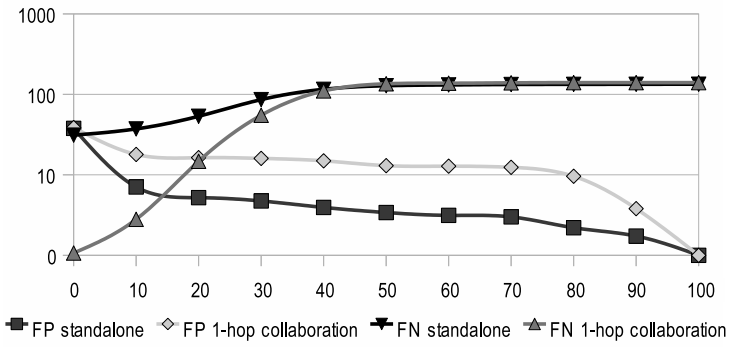
6.3 Hello flood

The hello flood attacker was implemented as a node whose signal strength is 40 dB stronger than it would be if a normal node was situated at the same place in the network.

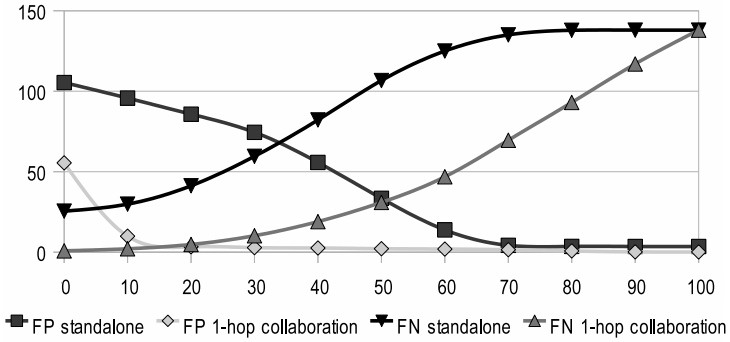
Relationship between the number of false positives (and negatives) and the signal strength detection threshold (δ_{SS}) can be seen in Figure 6a for standalone and 2-hop collaborating IDS agents. When collaboration was used and δ_{SS} was set to 5 dB (the approximate value at which the false positives and negatives curves intersect), 0.28 false positives and 3.76 false negatives were announced by the detection component on average in every detection round. Each of the five attackers was reported in approximately 223.23 alerts per detection round (see Figure 7a for more details). For the confrontation with the standalone IDS, it can be seen in Figure 6a that false positives and negatives



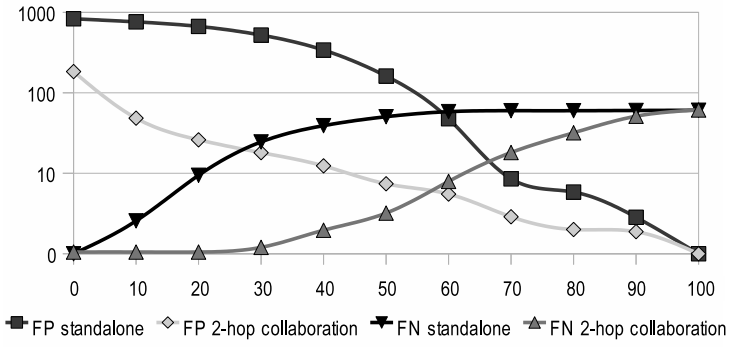
(a) hello flood attack



(b) selective forwarding attack



(c) deceptive and random jamming



(d) reactive jamming

Figure 6: Relationship between the number of false positives (FP) and negatives (FN), the Y axis, and the value of the detection threshold, the X axis.

curves intersect approximately at the value of δ_{SS} equal to 9 dB. In such case the IDS would report 153.28 false positives and 162.4 false negatives on average in every detection round.

In conclusion, the collaboration component is clearly helpful in finding true information about the attackers (their high *SS*). Hello flood attackers can be efficiently revealed using collaborating neighbor-based IDS.

6.4 Selective forwarding

Only nodes which serve as forwarders (internal nodes of the CTP tree) are able to conduct the selective forwarding attack. In our simulations they were chosen randomly from the subgroups of nodes that forwarded at least 300 messages during the very first 304 seconds. The required number of forwarded messages was chosen empirically and it ensured that there had been enough nodes to choose from (21 nodes on average). And most importantly, these nodes could be considered steady forwarders which would have not tended to become leaf nodes later during the simulation. Selective forwarders dropped every other packet in our implementation. γ_{PD} was set to value 19 (the IDS announced up to this number of dropped packets on average about non-malicious nodes during the simulations).

Relation among the number of false positives, false negatives and the packet dropping ratio threshold ($\delta_{\frac{PD}{PR'}}$) is depicted in Figure 6b for both standalone and 1-hop collaborating IDSs. As it can be seen there the number of false negatives was reasonably decreased by employing the collaboration module. Unfortunately, the number of false positives increased at the same time. When $\delta_{\frac{PD}{PR'}}$ was set to 21 (approximately the intersection of the 1-hop collaboration curves in Figure 6b), the standalone IDS reported 5.13 false positives (about 1.6 different nodes) and 55.47 false negatives on average in every detection round. Having the same value of $\delta_{\frac{PD}{PR'}}$ set and the collaboration component enabled, the number of false negatives decreased to 16.67. At the same time, the number of false positives was 16.33 on average in every detection round. However, the IDS marked only 1.27 different non-malicious nodes as selective forwarders by a mistake. Each of the attacker was revealed by approximately 15.65 and 24.65 IDS agents for the case of the standalone and 1-hop collaboration enabled system respectively (see Figure 7b for more details).

Both standalone and collaborating IDSs have their pros and cons when detecting selective forwarding. Although the number of false positives was high in the case of the

collaborating system, the number of different non-malicious nodes reported by these alerts was even lower than in the case of the standalone system. It can be concluded that our 1-hop collaborating IDS is capable of revealing the selective forwarding attack.

6.5 Deceptive and random jamming

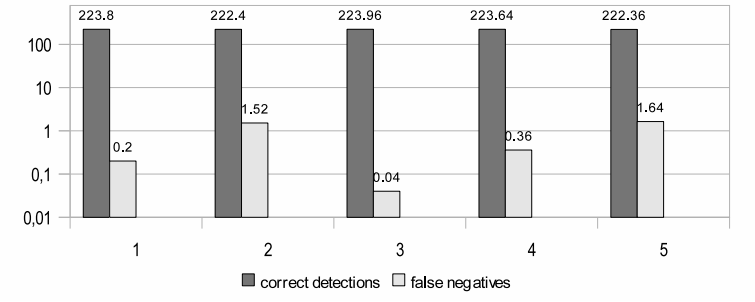
The deceptive jammer was implemented as a node which injects packets with no break regardless the medium access protocol. It is easy to accurately reveal such attacker as the amount of sent packets is extremely high in comparison to the normal node's PS. The results from simulations concerned with such deceptive jammer can be found in [2]. In [2], the deceptive jammer was identified based on its high PS.

In order to detect random jammer in this work received-to-sent ratio is monitored as the attacker sends some "extra" packets (which it did not receive to forward). The attacker has lower received-to-sent ratio than it is common in its neighborhood then. Obviously, the deceptive jammer can be revealed using this method too.

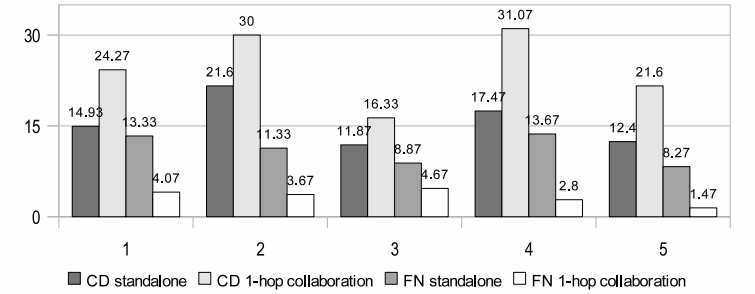
The random jammer is implemented as a node which sends packets randomly in the interval between 0.28 and 0.48 seconds. Hence, it sends from 633 to 1085 packets per τ (in comparison, approximately 16.76 out of 224 nodes send more than 700 packets and 6.56 of them send more than 1100 packets per τ). γ_{PS} was set to 699. We believe that node which sends less then 699 packets per detection period is not able to cause jamming (as non-malicious nodes tend to send this amount of packets too).

The results of the measurements of false positives and negatives can be seen in Figure 6c. The 1-hop collaborating IDS provided optimum results when the received-to-sent ratio threshold ($\delta_{\frac{PR}{PS}}$) was set to 15 (approximately the intersection of the false positives and negatives curves). This configuration was used and effectiveness of revealing deployed attackers is depicted in Figure 7c. The number of false positives was approximately 4.24 (2.8 different nodes) and there were 3.36 false negatives on average in every detection round. Each of the attacker was correctly detected by 26.93 IDS agents on average. The results obtained from the simulations in the standalone IDS configuration are not satisfactory as it can be seen in Figure 6c. This figure shows high number of false positives (90.64) and negatives (33.96) for this setting of the IDS.

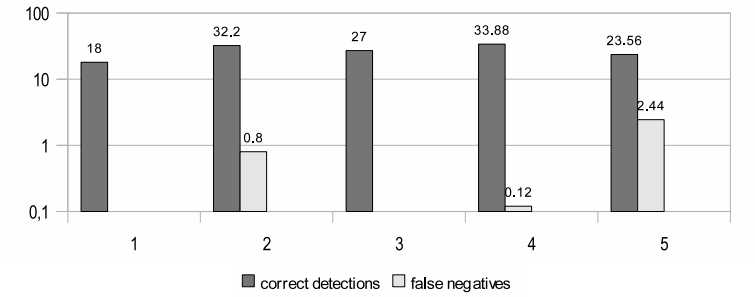
Eventually, the results suggest that our IDS with the collaboration component enabled can be used to detect deceptive and random jammers.



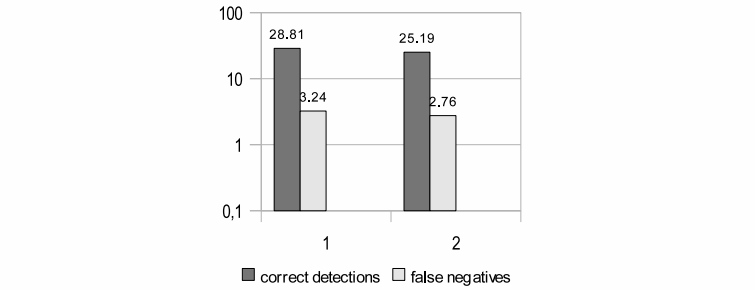
(a) hello flood attack (2-hop collaboration)



(b) selective forwarding (standalone and 1-hop collaboration)



(c) deceptive and random jamming (1-hop collaboration)



(d) reactive jamming (2-hop collaboration)

Figure 7: Comparison of the number of correct detections (CD) and false negatives (FN), the Y axis, for each of the 5 attackers (2 victims in the case of reactive jamming), the X axis.

6.6 Reactive jamming

The reactive jammer was not implemented in this work. However, the simulations were run in such manner so that some nodes' communication was jammed by some number of other nodes. Two nodes in each simulation was unable to deliver their messages. These simulations can be looked at as that there were ongoing reactive jamming attacks in the network.

It should be noted that the method described in Section 5.3 for revealing reactive jamming is not intended to reveal the jammer itself but only that the attack is ongoing. The method provides the information which node's neighbor is jammed.

Detection of reactive jamming was evaluated using both standalone and 1-hop and 2-hop collaborating neighbor-based IDSs. $\gamma_{PS-RETRY}$ was set to 49 (as the maximum number of retransmissions is 30 in the CTP and a node sends 19 packets per τ). The best results were obtained when 2-hop collaboration was enabled. As it can be seen in Figure 6d 2-hop collaborating IDS reported 5.96 false positives about 3.04 different nodes and 6 false negatives on average in every detection round. $\delta_{\frac{PA}{PS}}$ was set to 57 (see the approximate intersection of false positives and negatives curves in 6d). At the same time, there were 28.81 and 25.19 correct alerts reporting the two jammed nodes (see Figure 7d). On the other hand the standalone IDS was not able to cope with the detection of reactive jamming as it reported too many false positives (73.88) and false negatives (56.36) in this configuration.

These results suggest that our IDS with the collaboration module enabled for information exchange of 2-hop neighbors can be used to identify reactive jamming in the network.

7 Conclusion

We explored the jamming, hello flood, selective forwarding, sinkhole, sybil and packet alteration attacks in order to find out if they can be detected using the neighbor-based detection technique. Several symptoms of attacker's and victim's behavior were described and the reasons why they could or could not be used in a neighbor-based IDS were explained (see Section 3). We concluded that the received signal strength, packet dropping rate, packet sending rate and packet delivery ratio can be used to detect hello flood, selective forwarding and jamming attacks.

The earlier works are based on the assumption that in any local area all attribute vectors follow the same multivariate normal distribution. However, they do not provide any notion of circumstances under which the assumption holds (see Section 2). We conducted an experiment and revealed that the assumption did not hold for the packet dropping, receiving and sending rates in networks running a tree based routing protocol without data aggregation (see Appendix A).

The main problem we encountered while trying to apply the neighbor-based detection technique to networks without data aggregation was that even for nodes spatially close to each other packet receiving rates differ significantly (and packet sending, dropping and forwarding rates are implied by their values). Although the packet dropping rate and packet sending rate can be used in the neighbor-based technique for detection of selective forwarding and random jamming attacks respectively (see Sections 3.3 and 3.1), they will result into a high number of false positives and false negatives in networks operating tree-based routing protocols without data aggregation (see Appendix B). In Section 4 we proposed the statistics which mitigated the mentioned problem to an acceptable level for such networks. An IDS for the TinyOS operating system which uses packet delivery ratio and received signal strength as well as the proposed packet dropping ratio and received-to-sent ratio to detect reactive jamming, hello flood, selective forwarding and random jamming attacks respectively was implemented. We evaluated the IDS in the TOSSIM simulator. We found out that our IDS is capable of detecting all of the attacks mentioned above with reasonably low occurrence of false positives and negatives when collaboration among nodes is employed.

Acknowledgment

This work was supported by the project 102/09/H042 "Mathematical and Engineering Approaches to Developing Reliable and Secure Concurrent and Distributed Computer Systems" of the Czech Science Foundation.

References

[1] M. Demirbas and Y. Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *WOWMOM '06: Proceedings of the IEEE International Sym-*

posium on a World of Wireless Mobile and Multimedia Networks, pages 564–570, Washington DC, USA, 2006. IEEE Computer Society.

[2] L. Folkman. Neighbour-based intrusion detection in wireless sensor networks. Master's thesis, Masaryk University, Brno, Czech Republic, 2010.

[3] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. The collection tree protocol. In *SenSys '09: Proceedings of the ACM Conference on Embedded Networked Sensor Systems*, pages 1–14, 2009.

[4] T. Hai and E. Huh. Detecting selective forwarding attacks in wireless sensor networks using two-hops neighbor knowledge. In *NCA '08: Proceedings of the IEEE International Symposium on Network Computing and Applications*, pages 325–331, Washington, DC, USA, 2008. IEEE Computer Society.

[5] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *ASPLOS-IX: Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 93–104. ACM, 2000.

[6] L. Honus. Design, implementation and simulation of intrusion detection system for wireless sensor networks. Master's thesis, Masaryk University, Brno, Czech Republic, 2009.

[7] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, 1(2-3):293–315, 2003.

[8] I. Krontiris, T. Dimitriou, and F. C. Freiling. Towards intrusion detection in wireless sensor networks. 13th European Wireless Conference, 2007.

[9] I. Krontiris, T. Dimitriou, T. Giannetsos, and M. Mpasoukos. Intrusion detection of sinkhole attacks in wireless sensor ntworks. In *ALGOSENSORS '07: Proceedings of the International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, pages 150–161, Germany, 2007. Springer-Verlag.

[10] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *SenSys '03: Proceedings of the ACM Conference on Embedded Networked Sensor Systems*, pages 126–137, New York, NY, USA, 2003. ACM.

[11] G. Li, J. He, and Y. Fu. A group-based intrusion detection scheme in wireless sensor networks. In *GPC-WORKSHOPS '08: Proceedings of the International Conference on Grid and Pervasive Computing - Workshops*, pages 286–291, Washington, DC, USA, 2008. IEEE Computer Society.

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

[13] R. Roman, J. Lopez, and S. Gritzalis. Situation awareness mechanisms for wireless sensor networks. *IEEE Communications Magazine*, 46(4):102–107, April 2008.

[14] A. Stetsko and V. Matyas. Effectiveness metrics for intrusion detection in wireless sensor networks. In *EC2ND 2009: European Conference on Computer Network Defense. Milan, Italy. To appear in IEEE Computer Society*, 2009.

[15] M. Wen, H. Li, Y. Zheng, and K. Chen. Tdoa-based sybil attack detection scheme for wireless sensor networks. *Journal of Shanghai University (English Edition)*, 12:66–70, 2008.

[16] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: attack and defense strategies. *IEEE Network Magazine*, 20(3):41–47, 2006.

[17] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *MobiHoc '05: Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 46–57, New York, NY, USA, 2005. ACM.

A Analysis of distribution of statistics

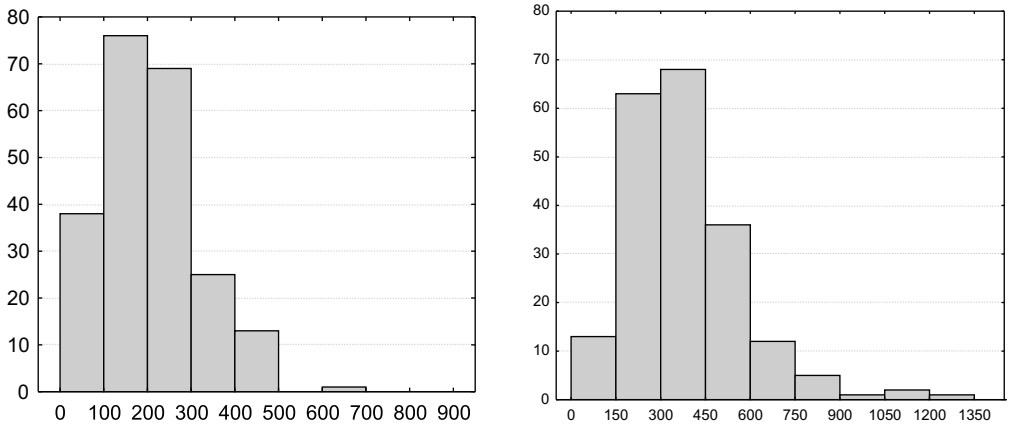
In this section the analysis of distribution of the packet sending, packet dropping and packet receiving rates as well as received signal strength is provided for a network operating a tree-based routing protocol without data aggregation. The simulation setup was the same as described in Section 6.2.

There were obtained 224 attribute vectors $F(a_i)$ $(i = \overline{1,224})$ of the form $\{\{f_k(b_{i1})\},...,\{f_k(b_{im_i})\}\}$ for different functions f_k, namely for the packet dropping rate, packet sending rate, packet receiving rate and received signal strength. The number of neighbors $m_i, i = \overline{1,224}$ varied from 8 to 100 (see Section 6.2). A node had 28 neighbors on average (median and standard deviation were 26 and 13 respectively).

Should a sample follow the normal distribution, its skewness and kurtosis must be within an expected range. The standard error of the skewness is $\sqrt{6/n}$ whereas the standard error of the kurtosis is $\sqrt{24/n}$. In both cases n is the size of the sample (attribute vector). If absolute values of either skewness or kurtosis of the sample are twice as big as the corresponding standard errors, we can assume that the sample does significantly differ from the normal distribution. When we refer to expected values of the skewness or kurtosis for a certain sample, we mean doubled corresponding standard errors for that sample. For each sample its expected values of the skewness and kurtosis were calculated. On average the expected skewness (kurtosis) value was 1 (2). The median was 0.96 (1.92) and the standard deviation was 0.22 (0.44).

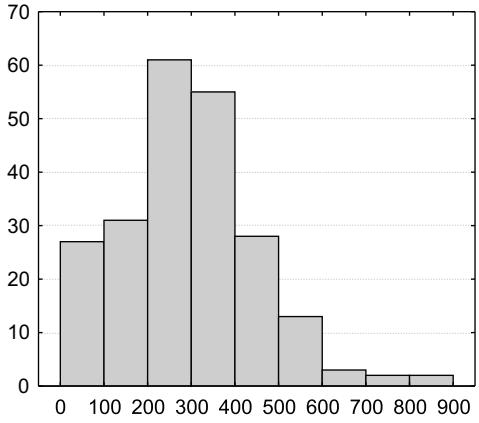
For the packet sending rate, 24 samples had the kurtosis value within the expected range. However, only two of them had acceptable skewness values. For each sample the difference (in percents) between its expected and actual values of the skewness was calculated. The following formule $|\frac{|S|-2*SES}{2*SES}|$ was used where S denotes the actual value of the skewness for a certain sample and SES denotes the standard error of the skewness for that sample. The distribution of the differences is depicted in Figure 8a. Two "exemplary" samples were removed from the distribution. The majority of the samples had their skewness values 100% bigger than the expected values.

For the packet dropping rate, 3 samples had the kurtosis value within the expected range. However, none of them had acceptable skewness values. Moreover, 23 samples contained only zeros. For such samples their skewness and kurtosis values cannot be calculated. These samples were removed from the set. The distribution of the dif-

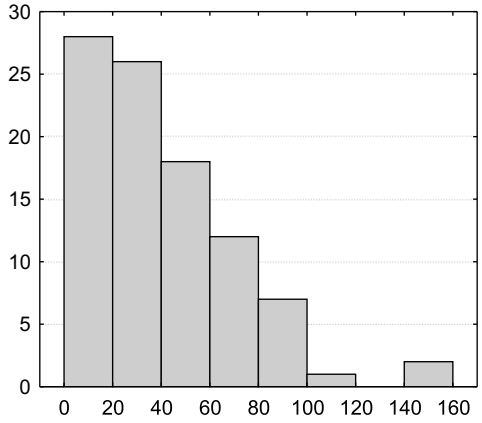


(a) packet sending rate

(b) packet dropping rate



(c) packet receiving rate



(d) received signal strength

Figure 8: Histograms. The X axis corresponds to the difference (in percents) between the expected value of skewness and the calculated one. The Y axis corresponds to the frequency of occurrence.

ferences (in percents) between the expected and calculated values of the skewness is depicted in Figure 8b.

For the packet receiving rate, 9 samples had the kurtosis value within the expected range. However, only 2 of them had acceptable skewness values. These 2 samples were not represented in the distribution of the differences (in percents) depicted in Figure 8c.

For the received signal strength, 218 samples had the kurtosis value within the expected range and 130 of them had acceptable skewness values. These 120 samples were not represented in the distribution of the differences (in percents) depicted in Figure 8d.

Based on the obtained results we concluded that the samples of packet sending, packet dropping and packet receiving rates are highly skewed right and "peaked" and do not follow the normal distribution.

B Analysis of accuracy of statistics

In this section we present the analysis on how the choice of the statistic affects the accuracy of the IDS. The analysis was done using the TOSSIM simulator. The simulation setup was the same as described in Section 6.2.

B.1 Selective forwarding attack

There were 10 selective forwarders that were chosen randomly from the group of nodes that had forwarded at least 300 messages during the very first 304 seconds. An attacker was dropping from 20 to 50 percents of packets. For detection of the selective forwarding attack both PD (see Section 3.3) and $\frac{PD}{PR'}$ (see Section 4) statistics can be used:

$$\frac{PD(b_{ij})}{PR'(b_{ij})} - AVG(\frac{PD(b_{i1})}{PR'(b_{i1})}, ..., \frac{PD(b_{im_i})}{PR'(b_{im_i})}) > \delta_{\frac{PD}{PR'}} \wedge PD(b_{ij}) > 19 \tag{10}$$

$$PD(b_{ij}) - AVG(PD(b_{i1}), ..., PD(b_{im_i})) > \delta_{PD} \wedge PD(b_{ij}) > 19 \tag{11}$$

For both Rules 10 and 11 we found the ranges of $\delta_{\frac{PD}{PR'}}$ and δ_{PD} such that the minimum value of a range resulted into the highest (lowest) number of false positives (false negatives) while the maximum value resulted into the lowest (highest) number of false positives (false negatives) for a corresponding rule. The maximum number of false negatives is the same for both rules as well as the maximum number of false positives, the minimum number of false negatives and the minimum number of false positives.

Both standalone and collaborative IDSs employing Rule 10 were more accurate than the corresponding IDSs employing Rule 11 (see Figure 9).

Standalone mode. For $\delta_{PD} = 24.5$ (approximately the interaction of the "FP PD" and "FN PD" curves in Figure 9a) the number of false positives was 98 and the number of false negatives was 93. For $\delta_{\frac{PD}{PR'}} = 3.2$ (approximately the interaction of the "FP PD/PR" and "FN PD/PR" curves) the number of false positives was 80 and the number of false negatives was 77.

Collaborative mode. For $\delta_{PD} = 50.4$ (approximately the interaction of the "FP PD" and "FN PD" curves in Figure 9b) the number of false positives was 33 and the number of false negatives was 32. For $\delta_{\frac{PD}{PR'}} = 14.8$ (approximately the interaction of the "FP PD/PR" and "FN PD/PR" curves) the number of false positives was 8 and the number of false negatives was 9.

B.2 Random jamming attack

There were five random jammers in the network. An attacker was implemented as a node which sent packets randomly in the interval between 0.28 and 0.48 seconds. Hence, it sent from 633 to 1085 packets per detection round. For detection of the random jammers both PS (see Section 3.1) and $\frac{PR}{PS}$ (see Section 4) statistics can be used:

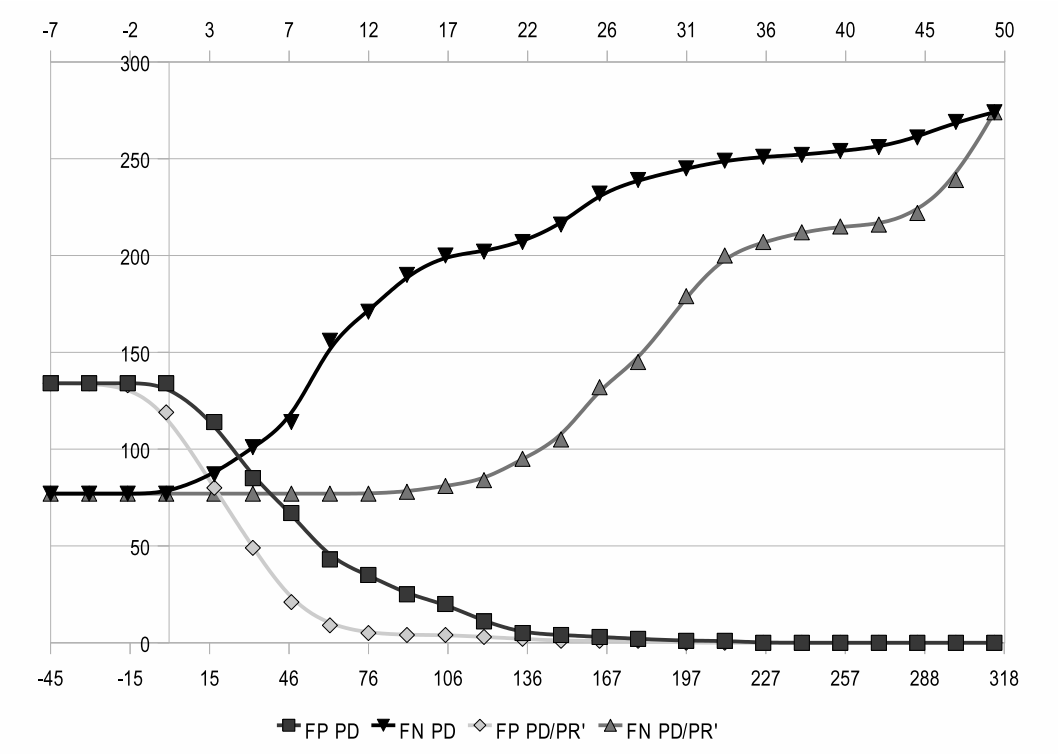
$$AVG(\frac{PR(b_{i1})}{PS(b_{i1})}, ..., \frac{PR(b_{im_i})}{PS(b_{im_i})}) - \frac{PR(b_{ij})}{PS(b_{ij})} > \delta_{\frac{PR}{PS}} \wedge PS(b_{ij}) > 699 \quad (12)$$

$$PS(b_{ij}) - AVG(PS(b_{i1}), ..., PS(b_{im_i})) > \delta_{PS} \wedge PS(b_{ij}) > 699 \quad (13)$$

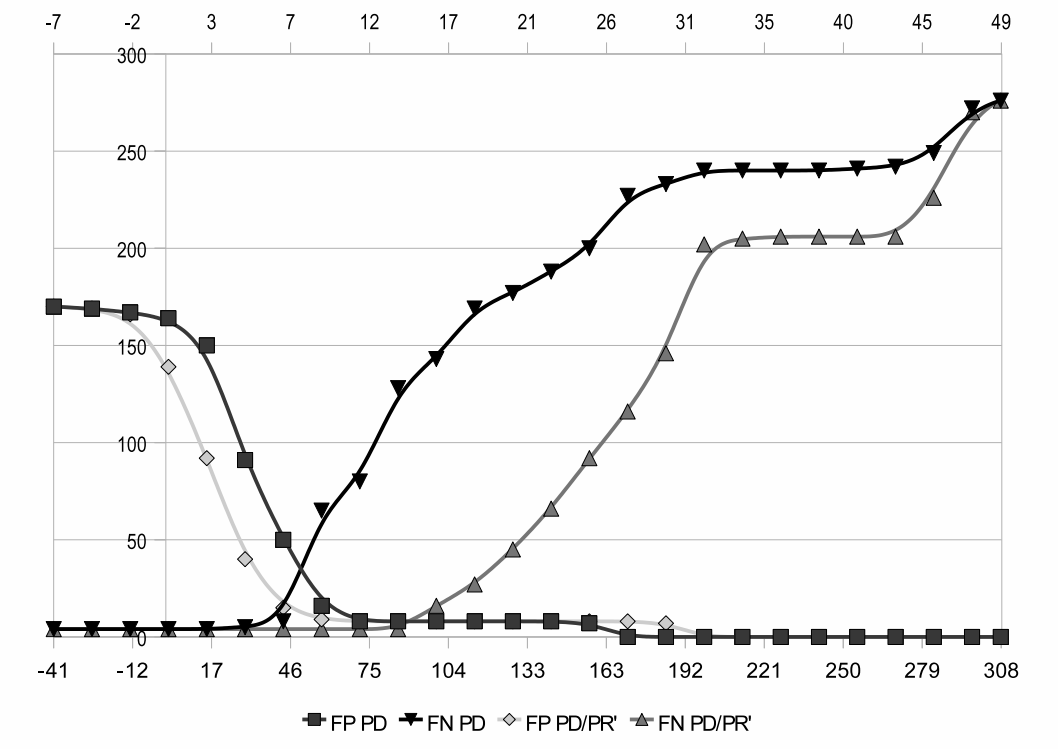
Both standalone and collaborative IDSs employing Rule 12 were more accurate than corresponding IDSs employing Rule 13 (see Figure 10).

Standalone mode. For $\delta_{PS} = 870$ (approximately the interaction of the "FP PS" and "FN PS" curves in Figure 10a) the number of false positives was 135 and the number of false negatives was 137. For $\delta_{\frac{PR}{PS}} = 56.2$ (approximately the interaction of the "FP PR/PS" and "FN PR/PS" curves) the number of false positives was 30 and the number of false negatives was 31.

Collaborative mode. For $\delta_{PS} = 923$ (approximately the interaction of the "FP PS" and "FN PS" curves in Figure 10b) the number of false positives was 131 and the number of false negatives was 137. For $\delta_{\frac{PR}{PS}} = 43.37$ (approximately the interaction of the "FP PR/PS" and "FN PR/PS" curves) there were only one false positive and one false negative.

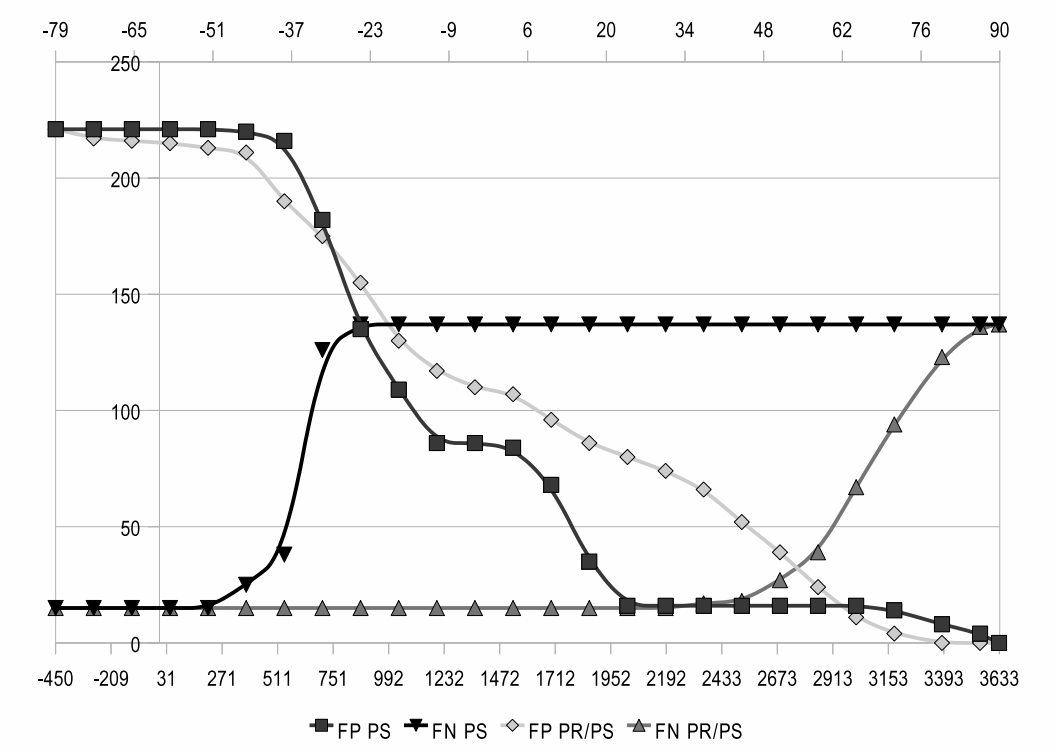


(a) standalone

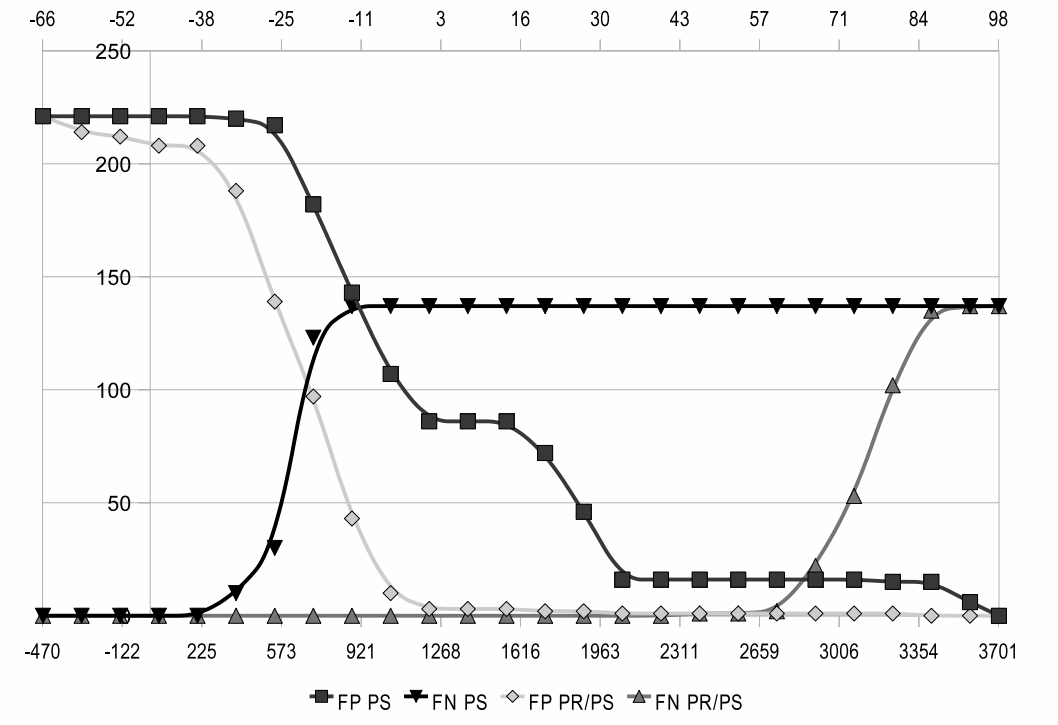


(b) 1-hop collaboration

Figure 9: Accuracy of PD and $\frac{\text{PD}}{\text{PR}'}$ statistics in detection of the selective forwarding attack. The bottom X axis corresponds to δ_{PD}, while the top X axis corresponds to $\delta_{\frac{\text{PD}}{\text{PR}'}}$. The Y axis represents the number of false positives (FP) and false negatives (FN).



(a) standalone



(b) 1-hop collaboration

Figure 10: Accuracy of PS and $\frac{\text{PR}}{\text{PS}}$ statistics in detection of the random jamming attack. The bottom X axis corresponds to δ_{PS}, while the top X axis corresponds to $\delta_{\frac{\text{PR}}{\text{PS}}}$. The Y axis represents the number of false positives (FP) and false negatives (FN).