WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2015; **15**:276–294
Published online 28 January 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.2341

RESEARCH ARTICLE

Anomaly-based intrusion detection of jamming attacks, local versus collaborative detection

Alexandros G. Fragkiadakis*, Vasilios A. Siris†, Nikolaos E. Petroulakis and Apostolos P. Traganitis

Institute of Computer Science, Foundation for Research and Technology–Hellas, PO Box 1385, GR 711 10 Heraklion, Crete, Greece

ABSTRACT

We present intrusion detection algorithms to detect physical layer jamming attacks in wireless networks. We compare the performance of local algorithms on the basis of the signal-to-interference-plus-noise ratio (SINR) executing independently at several monitors, with a collaborative detection algorithm that fuses the outputs provided by these algorithms. The local algorithms fall into two categories: simple threshold that raise an alarm if the output of the SINR-based metrics we consider deviates from a predefined detection threshold and cumulative sum (cusum) algorithms that raise an alarm if the aggregated output exceeds the predefined threshold. For collaborative detection, we use the Dempster–Shafer theory of evidence algorithm. We collect SINR traces from a real IEEE 802.11 network, and with the use of a new evaluation method, we evaluate both the local and the Dempster–Shafer algorithms in terms of the detection probability, false alarm rate, and their robustness to different detection threshold values, under different attack intensities. The evaluation shows that the cusums achieve higher performance than the simple threshold algorithms under all attack intensities. The Dempster–Shafer algorithm when combined with the simple algorithms, it can increase their performance by more than 80%, but for the cusum algorithms it does not substantially improve their already high performance.

KEYWORDS

collaborative intrusion detection; signal-to-interference-plus-noise ratio; jamming; simple threshold algorithms; cumulative sum algorithms; performance evaluation; Dempster–Shafer theory of evidence

***Correspondence**

Alexandros G. Fragkiadakis, Institute of Computer Science, Foundation for Research and Technology–Hellas, PO Box 1385, GR 711 10 Heraklion, Crete, Greece.
E-mail: alfrag@ics.forth.gr

1. INTRODUCTION

The deployment of wireless networks has offered inexpensive, convenient, and ubiquitous access to millions of users through access points located at numerous public places (e.g., airports and university campuses). However, their broadcast nature makes them highly susceptible to attacks. Attackers can exploit vulnerabilities in the medium access control and physical layers and heavily disrupt the services between the network nodes (e.g., see [1–6]). For these reasons, intrusion detection is a primary concern within the research community.

In general, intrusion detection algorithms fall into two categories: misuse (or signature-based) detection and anomaly-based detection. The former is based on known signature attacks, it has low false alarm rates, but it lacks the ability to detect new types of attacks. The latter may have higher false alarm rates, but it has the potential ability to detect unknown types of attacks. In this work, we study the detection of physical layer jamming attacks; therefore, we believe that anomaly-based detection is better suited to detect this type of attack as our associated metrics are based on signal-to-interference-plus-noise ratio (SINR) that is highly volatile.

We use a periodic attacker (referred as Jammer throughout the paper) that emits energy on a neighboring channel legitimates nodes use for communication. Following this jamming model, the attacks become feasible through the generation of interference, and their detection is performed using SINR-based metrics. Other types of jamming (e.g., on the same channel) and the associated detection techniques are described in [7]. However, these attacks

†Vasilios A. Siris is also with the Department of Informatics, of the Athens University of Economics and Business, Greece.

are out of the scope of this work as we focus on the physical layer.

Our intrusion detection algorithms are of two types: local detection algorithms and a fusion algorithm. The former execute independently in a number of monitors seeking for changes in the statistical characteristics of the SINR that include the average SINR, the minimum SINR, and the maximum-minus-minimum SINR, in a short window. The latter fuses the outputs provided by the local algorithms, thus forming a distributed collaborative intrusion detection system. The fusion algorithm we investigate is the Dempster–Shafer theory of evidence (DS) [8]. Moreover, our contribution for combining measurements is based on the outputs of the local detection algorithms without the need to transmit SINR values in a per-packet basis, opposed to other contributions (e.g., [9]). Also, we use a new method for the evaluation of the proposed algorithms based on score assignments and a new concept to define and study robustness issues under different detection threshold values.

Our main contributions are as follows:

- We consider two types of local detection algorithms: simple threshold and cumulative sum (cusum).
- We consider different metrics for the local algorithms based on the SINR: average, minimum, and maximum-minus-minimum SINR.
- We consider collaborative detection to improve local algorithms' performance.
- We use the term robustness to describe the algorithms' performance stability under different detection threshold values.
- We investigate the performance of the local and the fusion algorithms in terms of the detection probability, false alarm rate, and their robustness to different detection threshold values.
- We present the performance of the local and fusion algorithms considering measurements from a real network, under two attack intensities, collected from locations at various distances from the Jammer (we repeat the experiments placing the Jammer in a different location).

The evaluation shows that the cusum algorithms achieve, in general, higher performance than the simple ones. Especially, the cusum algorithm that considers the maximum-minus-minimum SINR metric has superior performance in all scenarios. When the fusion algorithm is used, the performance of the simple algorithms substantially increases, whereas when combined with the cusums, performance still remains high.

The remainder of this paper is organized as follows. In Section 2, related work is presented. In Section 3, we describe the experimental layout and the mechanism used to collect the SINR measurements. Section 4 presents the jamming model used to launch the attacks. The description of the local detection algorithms is given in Section 5. The definition of robustness is given in Section 6. The method we use to set the parameters of the detection algorithms is given in Section 7. In Section 8, we present the performance evaluation of the local algorithms. Section 9 presents our collaborative intrusion detection system and the use and evaluation of the DS. Finally, conclusions and further work appear in Section 10.

2. RELATED WORK

There is extended research on the detection of attacks in wireless networks with important contributions. In [10], a distributed system for intrusion detection is described that executes in every wireless node. Events are locally generated (e.g., packet transmission/reception and frame type) by every node and then sent to a single fusion center (FC). FC then, on the basis of majority voting, tries to detect intrusions. The authors claim that the proposed work can detect attacks at the physical and medium access (MAC) layers. However, they do not provide any evaluation results regarding the detection probability and the false alarm rate. Furthermore, as every event is recorded and sent to FC, a high volume of control traffic is generated that can negatively affect network's performance. Moreover, every node has to take part in this scheme that is however not realistic in real wireless implementations. On the contrary, our work uses dedicated monitors for intrusion detection suppressing control traffic overhead by communicating with the FC only when an alarm is locally signaled.

Fusion center fingerprinting is proposed in [11] for the detection of MAC spoofing. This is a multisensor system that, on the basis of the radio frequency (RF) fingerprints created at a number of sensors, tries to detect rogue devices that have spoofed their MAC addresses. This system uses physical layer features to detect MAC layer misbehavior, whereas we use SINR to detect jamming at the physical layer. MAC misbehavior is also addressed in [6], where the authors consider the sequential probability ratio test. Wood *et al*. [12] proposed DEEJAM, a MAC layer protocol for defending against stealthy jammers using IEEE 802.15.4-based hardware. Nevertheless, as the authors noted, against a powerful and more sophisticated Jammer, DEEJAM cannot effectively defend the wireless network.

In [13], two types of algorithms are described for the detection of SYN attacks. Their evaluation shows that the simple detection algorithm has satisfactory performance for the high-intensity attacks, but it deteriorates for the low-intensity ones. On the other hand, the cusum algorithm has robust performance for the different types of attacks. This is consistent with the findings of our work, although we perform measurements at the physical layer.

In [14], methods for anomaly detection and distributed intrusion detection in mobile ad hoc networks are proposed focusing on two routing protocols. The authors use a two-layer hierarchical system where anomaly indexes are combined using an averaging or median scheme. The evaluation results show that the averaging scheme achieves higher performance.

Peng *et al*. [15] proposed a cusum algorithm used to collect statistics at local systems, whereas a learning algorithm decides about when information has to be shared among the nodes to minimize detection delay and reduce the communication overhead. In this work, data are fused using the sum rule.

In [16], the authors described a distributed change point detection scheme for the detection of distributed denial-of-service attacks over multiple network domains. At each router, a cusum algorithm executes raising alerts that are sent to a central server. Then, the server creates a subtree displaying a spatiotemporal vision of the attack. In a second hierarchy level, a global picture of the attack is created by merging all subtrees together.

Collaborative intrusion detection has been considered in several contributions such as in [17], where data provided by heterogeneous intrusion detection monitors are fused. The proposed scheme considers metrics for the detection of UDP and ICMP flooding attacks, as well as SYN attacks. The authors in [18] use DS to detect attacks by combining events generated by multiple layers. Each event based on metrics such as the received-signal-strength-indicator (RSSI) and time-to-live (TTL) is assigned with a belief, a measure of confidence about a specific attack. The beliefs are then fused by DS signaling a possible attack. Our work has two major differences: (i) we use DS to combine beliefs from multiple monitors and not from multiple layers of a single node, and (ii) our scheme detects attacks at the physical layer and not higher-layer attacks as these discussed in this related work (e.g., man-in-the-middle attack). DS has also been considered in [19–22], however, for the detection of higher-layer attacks.

The so-far described related contributions focus on local, distributed, or collaborative schemes for the detection of attacks at higher network layers (e.g., internet and transport), whereas our work focuses on detecting attacks at the physical layer of a wireless network using SINR measurements. SINR is volatile and highly correlated to the location of a monitor, thus making intrusion detection based on SINR more challenging. Next, we describe several contributions that consider physical layer attack detection.

In [23], the authors described several types of jammers proposing two types of detection algorithms that consider metrics such as the *packet delivery ratio*, the *bad packet ratio*, and the *energy consumption amount*. The basic algorithm tries to detect jamming by using multiple if-else statements on the aforementioned metrics, whereas the advanced algorithm uses a distribution scheme where information is collected from neighboring nodes. The evaluation shows high detection rates, but trade-offs regarding the false alarm rate versus the detection probability or the robustness of the algorithms are not presented.

Techniques that detect anomalies at all layers of a wireless sensor network are proposed in [24]. The authors showed how the detection probability increases when the number of the nodes running the proposed procedure increases, but they did not show the trade-off with the false alarm rate.

In [9] is shown how the errors at the physical layer propagate up the network stack, presenting a distributed anomaly detection system based on simple thresholds. A method for combining measurements using the Pearson's product moment correlation coefficient is also presented. A disadvantage of this method is that *raw* RSSI measurements by several sniffers are needed. This could generate a high volume of traffic flowing from the sniffers to a main node where the algorithm executes. In contrast, our proposal is based on the outputs of several local detection algorithms without the need of transmitting SINR values in a per-packet basis, thus saving valuable resources.

Several adversarial models are presented in [25], all focusing on RF jamming attacks. One of the proposed algorithms applies *high-order crossings*, a spectral discrimination mechanism that distinguishes normal scenarios from two types of the defined jammers. The authors introduced two detection algorithms based on thresholds that use signal strength and location information as a consistency check to avoid false alarms.

The authors in [26] presented a cross-layer approach to detect jamming attacks. Jamming is performed at the physical layer by using RF signals and at the MAC layer by targeting the request-to-send/clear-to-send (RTS/CTS) and network allocation vector (NAV) mechanisms of the IEEE 802.11 protocol. Jamming detection is split into two phases. In the first phase, simple threshold algorithms are deployed using metrics such as the physical carrier sensing time, the number of RTC/CTS frames, the duration of channel idle period, and the average number of retransmissions. The second phase is triggered in the case of threshold violations.

The authors in [27] described ARES, an antijamming reinforcement system for 802.11 networks that tunes the parameters of rate adaptation and power control to improve the performance in the presence of jammers. However, ARES has to be implemented in every wireless node to regulate rate and power, whereas our system consists of dedicated monitors performing passive measurements; thus, no modifications are needed for the wireless clients.

Although significant, none of these contributions investigates the robustness of the detection algorithms as we do in this work.

3. EXPERIMENTAL LAYOUT COLLECTION OF SIGNAL-TO-INTERFERENCE-PLUS-NOISE RATIO MEASUREMENTS

The algorithms investigated in this work are based on SINR traces collected from a real IEEE 802.11a experimental network configured in an ad hoc mode (Figure 1). All nodes are equipped with Mini-ITX boards carrying 512 MB of RAM and a 80 GB hard disk. Moreover, the boards are equipped with Atheros 802.11a/b/g CM9-GP mini-PCI cards, controlled by Ath5k, an open source IEEE 802.11 driver [28], on Gentoo Linux. UDP traffic is

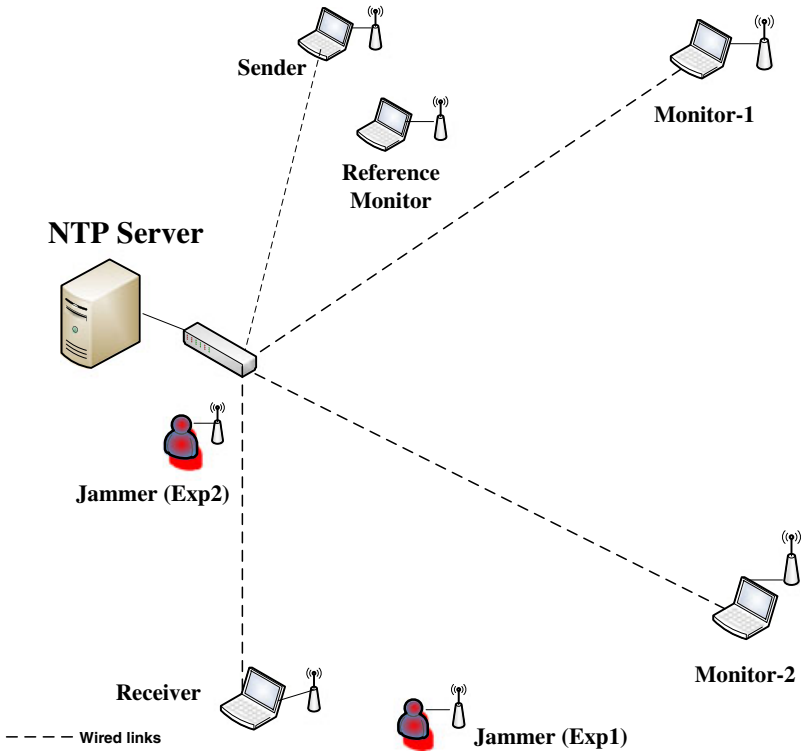


Figure 1. Network layout for the collection of signal-to-interference-plus-noise ratio measurements. NTP, network time protocol.

transmitted from Sender to Receiver (Figure 1) at a constant rate of 18 Mbps. We chose UDP as the transport protocol to focus on the jamming consequences, avoiding TCP's congestion control. We use a wired backbone network (denoted by the dotted lines in Figure 1) to control the experiments from a single machine, and time synchronization is achieved using a network time protocol server [29]. All nodes, except Jammer, operate on the same channel. Furthermore, we use two monitors with their interface cards set to monitor mode; hence, they receive all packets transmitted in the channel. Note that Jammer is placed at different locations for each experiment, as we conduct two experiments changing its position at each experiment. The Reference Monitor (RM) is used to assist in the definitions of the two attack intensities (high and low) used in the experiments.

Receiver, Monitor-1, and Monitor-2 collect SINR measurements (in a per-packet basis) by using a modified version of the Ath5k driver. The software layout for the collection of the SINR values is shown in Figure 2. These are collected along with their corresponding timestamps. The timestamps are produced, as soon as a packet is received, using the time module of the Linux operating system, as they are necessary to time align measurements at all nodes participating in the experiment, making feasible the performance evaluation of the algorithms afterwards. The modification of the Ath5k driver not only enabled the collection of the SINR traces at all nodes but also made feasible the collection of additional information at a per-packet basis such as the long and short retry counters for the packet retransmissions, the retry counters regarding the clear channel assessment (CCA) mechanism, the retry field of the MAC header, port numbers and IP addresses, and the timestamps of the outgoing or the incoming packets. Our software collection module (Figure 2) consists of several parts laying on both the kernel and user spaces of the Linux operating system. Data are collected through Ath5k in the kernel space and then asynchronously transmitted to the user space through the netlink socket interface. In the user space, the reception thread receives the data from the kernel space and stores them in a first-in–first-out (FIFO) queue. The storage thread pulls out the data from the queue and stores them in the hard disk. The reason for the user space functionality to be split into the aforementioned threads, connected through the queue, is to increase performance because the copy of the data from memory to the hard disk should not block the reception of the new data coming from the kernel space. The storage and reception threads execute independently in the multithreaded environment of Linux.

4. JAMMING MODEL

Generally, regarding jamming implementations, there is always the trade-off between jamming intelligence and

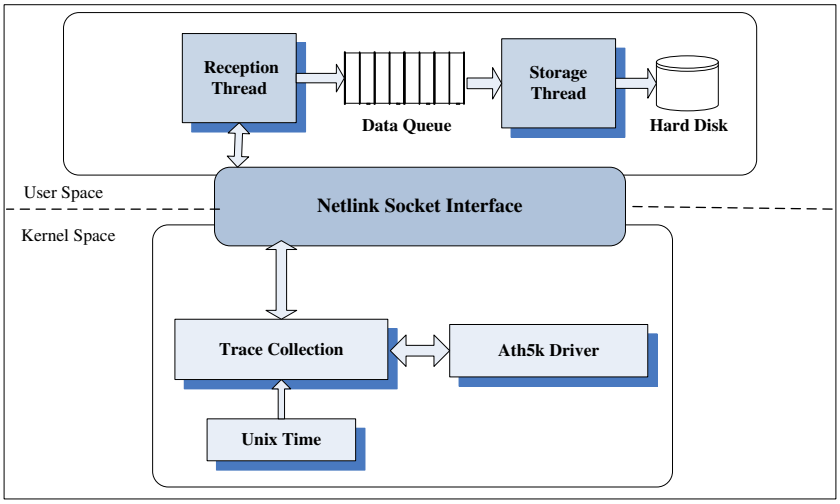


Figure 2. Measurement module based on the Ath5k driver.

cost. An intelligent Jammer can cause severe denial-of-service attacks with a low-energy consumption, but its cost can be significantly high (e.g., [30]). On the other hand, a less sophisticated Jammer, based on off-the-shelf hardware, can cause significant performance degradation, although consuming more energy, but it costs less and can also be used by individuals with any specialized knowledge on network protocols. The Jammer we use in this work is an off-the-shelf *periodic Jammer* that emits RF energy, alternating between sleeping and jamming. Jamming is performed at the physical layer by using a single node (Figure 1). This node operates on a neighboring channel to the one that the legitimate nodes communicate; thus, we perform jamming by generating interference. Jammer transmits UDP broadcast data at a constant bit rate of 6 Mbps. It transmits data for 30 s, after which it remains inactive for 30 s. It is placed in a location such that it is close to Receiver and *hidden* from Sender. Jammer was successfully hidden by the Sender by changing its (antenna) orientation (but without moving the jammer into a different location). Furthermore, several obstacles (e.g., furniture) helped on this, as the experiments were conducted in a working environment. Using this topology, Sender continuously transmits packets as it is unaware of Jammer's presence; consequently, its CCA mechanism is not activated. Furthermore, Jammer's driver after its modification, according to Figure 2, has extra operation characteristics so as its CCA and backoff mechanisms are disabled. This became feasible by modifying the values of several hardware registers that are part of the CM9-GP Atheros card. The advantage gained with the elimination of the CCA and backoff mechanisms is that Jammer is not affected by the transmissions of the legitimate nodes; thus, it can freely perform jamming.

We use two attack intensities (high and low) and two different series of experiments, referred as Exp_1 and Exp_2. Both experiments ran in the same building, and the only difference between them is Jammer's position (Figure 1) and the time they were conducted. Exp_1 executed late in the evening when most people had left work, whereas Exp_2 ran early in the noon during busy working hours. We executed the experiments to evaluate our intrusion detection algorithms by placing Jammer in different positions, as well as to study their performance in a busy working environment where the movements of people could possibly affect the SINR values measured by the monitors.

Furthermore, we define two types of attack intensities based on measurements of the packet loss and the throughput degradation at the Receiver and the noise power (interference) reported by the RM (Figure 1) that runs the Airmagnet WiFi Analyzer software [31]. The attack intensities are defined as follows:

- High-intensity attack, where the packet loss is over 50%, the throughput degradation over 80%, and the interference reported by RM is over −55 dbm.
- Low-intensity attack, where the packet loss is less than 15%, the throughput degradation less than 30%, and the interference reported by RM is below −75 dbm.

5. LOCAL DETECTION ALGORITHMS

The local detection algorithms execute independently at both monitors and the Receiver (Figure 1), falling into two categories: (i) simple threshold algorithms and (ii) cusum change point detection algorithms. Both types are applied to different metrics that are based on the SINR: average SINR, minimum SINR, and maximum-minus-minimum SINR. The values of the metrics are measured over a small time window and then, are compared with another metric over a large time window.

5.1. Simple threshold algorithms

The simple threshold algorithms trigger an alarm when the metric that the algorithm considers deviates from its normal (expected) value by some amount. The normal value is given by the value of the metric, estimated in some long time interval, whereas the degree of deviation to signal an alarm is determined by the detection threshold.

5.1.1. The simple min algorithm

The metric used by this algorithm is the minimum value of the SINR in a small window K. An alarm is raised if the minimum value of the SINR in the small window deviates from the average value of the SINR measured over a long window M. Let $SINR_n$ be the SINR value for frame (sample) n. If N is the number of samples, then for $n \in [M+1, N]$, the minimum SINR in the short window K is

$$SINR_{\min}(n) = \min_{n-K+1<i\leq n} SINR_i$$

whereas the average SINR measured in window M is

$$\overline{SINR}(n) = \frac{\sum_{i=n-M+1}^{n} SINR_i}{M}$$

An alarm is raised at the arrival of frame n if

$$\frac{\overline{SINR}(n)}{SINR_{\min}(n)} \geq h \tag{1}$$

where h is the detection threshold.

5.1.2. The simple max–min algorithm

Rather than considering the minimum value of the SINR measured in a small window, this algorithm considers the maximum-minus-minimum values of the SINR measured in that window. If D denotes the maximum-minus-minimum value of the SINR, then

$$D(n) = \max_{n-K+1<i\leq n} SINR_i - \min_{n-K+1<i\leq n} SINR_i$$

whereas the average value of D in the long window is

$$\bar{D}(n) = \frac{\sum_{i=n-M+1}^{n} D(i)}{M}$$

An alarm is raised at the arrival of frame n if

$$D(n) - \bar{D}(n) \geq h \tag{2}$$

5.1.3. The simple average algorithm

This algorithm compares the average value of the SINR in a short window, with the average value in a long window. If $\overline{SINR_{\text{short}}}(n) = \frac{\sum_{i=n-K+1}^{n} SINR_i}{K}$ is the average value of the SINR in a short window K, then an alarm is raised if

$$\frac{\overline{SINR}(n) - \overline{SINR_{\text{short}}}(n)}{\overline{SINR}(n)} \geq h \tag{3}$$

where $\overline{SINR}$ is the average SINR in the long window.

5.2. Cumulative sum algorithms

The second category of the local algorithms we investigate, is the cusum algorithm. This type of algorithm has been widely used in the literature [6,16,32–34]. In general, there are two main categories of cusum algorithms: (i) parametric and (ii) nonparametric. For the parametric cusum, a parametric model for $\{x\}$, where x is an independent and identically distributed random variable, is required, which is not easy to obtain in the area of the communication networks, and especially for the SINR because of its volatile nature. For this reason, we use nonparametric cusum algorithms where a model of $\{x\}$ (i.e., SINR) is not required. The cusum algorithms considered in this work are the cusum average (C_{avg}), cusum min ($C_{\min}$), and cusum max–min (C_{mm}).

5.2.1. The cusum min algorithm

The regression formula for the cusum min algorithm is given by

$$y_n = \begin{cases} y_{n-1} + Z_n - a & \text{if } y_n \geq 0 \\ 0 & \text{if } y_n < 0 \end{cases} \tag{4}$$

where $a > 0$ is a tuning parameter and $Z_n = \frac{\overline{SINR}(n)}{SINR_{\min}(n)}$. Note that y_n in (4) increases as $Z_n = \frac{\overline{SINR}(n)}{SINR_{\min}(n)} > a$, that is, as the minimum SINR value is smaller than the average SINR value by some amount that is determined by the value of a. In the experimental evaluation, we select $a = 0.7$ (in Section 7, we show how the optimal value of a is selected). An alarm is signaled when

$$y_n \geq h \tag{5}$$

where h is the detection threshold.

5.2.2. The cusum max–min algorithm

This algorithm has the same regression formula as the cusum min algorithm given by (4); however, Z_n is now given by

$$Z_n = D(n) - \bar{D}(n)$$

where $D(n)$ and $\bar{D}$ are the maximum-minus-minimum SINR in the short time window and the average maximum-minus-minimum SINR estimated in the long time window, respectively. For the experimental evaluation, we select $a = 8$. The alarm rule for the cusum max–min algorithm is identical to the rule for the cusum min algorithm given by (5).

5.2.3. The cusum average algorithm

As earlier, this algorithm has the same regression formula as the cusum min algorithm given by (4); however, Z_n is now given by

$$Z_n = \frac{\overline{SINR}(n) - \overline{SINR_{\text{short}}}(n)}{\overline{SINR}(n)}$$

For the experiments, we select $a = 0.8$. The alarm rule for the cusum max–min algorithm is identical to the rule for cusum min algorithm (5).

6. ROBUSTNESS AND THE PERFORMANCE EVALUATION METHOD

In this section, we describe the method we follow for evaluating the detection algorithms. Evaluation is performed in terms of the detection probability (*DP*), false alarm rate (*FAR*), and their robustness to different detection threshold values. Detection probability is defined as the ratio of the detected attacks over the total number of the attacks. The false alarm rate is the ratio of the number of false alarms over the total duration of the experiment (expressed in false alarms/minute). A false alarm occurs when there is no attack but an alarm is raised.

In most related works, performance evaluation is presented by showing the trade-off points between *FAR* and *DP* (e.g., [13,16,35–37]). As an example, Figure 3 shows the trade-off points for the local algorithms when the SINR traces collected at Monitor-1 during Exp_1 are considered. With this method, algorithms' performance increases when their associated trade-off points are closer to the left top corner of each graph (higher detection probability with a lower false alarm rate). Each trade-off point corresponds to a different detection threshold value. Although this is a significant method for performance evaluation, it is not complete as it provides no information regarding the robustness of the algorithms. By robustness, we mean how the performance in terms of the detection probability and false alarm rate varies, when the detection threshold changes. Moreover, this simplistic approach is not appropriate when the number of the algorithms under evaluation or the experimental data increase, as it is predicated on subjective criteria.

In this paper, we investigate algorithms' robustness along with the (traditional) *DP–FAR* evaluation. We vary the detection threshold from zero to a maximum value that an algorithm gives no alarms. Furthermore, we define as *score* $S \in R^+$ a number assigned to an algorithm based on its *DP–FAR* trade-off points. Score is given by $S = b * (c - d)$, where $d = \sqrt{FAR^2 + (1 - DP)^2}$ is the distance of a trade-off point (for a specific threshold h) from the optimum point ($DP = 1$ and $FAR = 0$) and $b, c \in \mathbb{R}^+$. For each *DP–FAR* pair, a different value of S is assigned (in [7], we analytically describe how S is computed). From all the available trade-off points, we

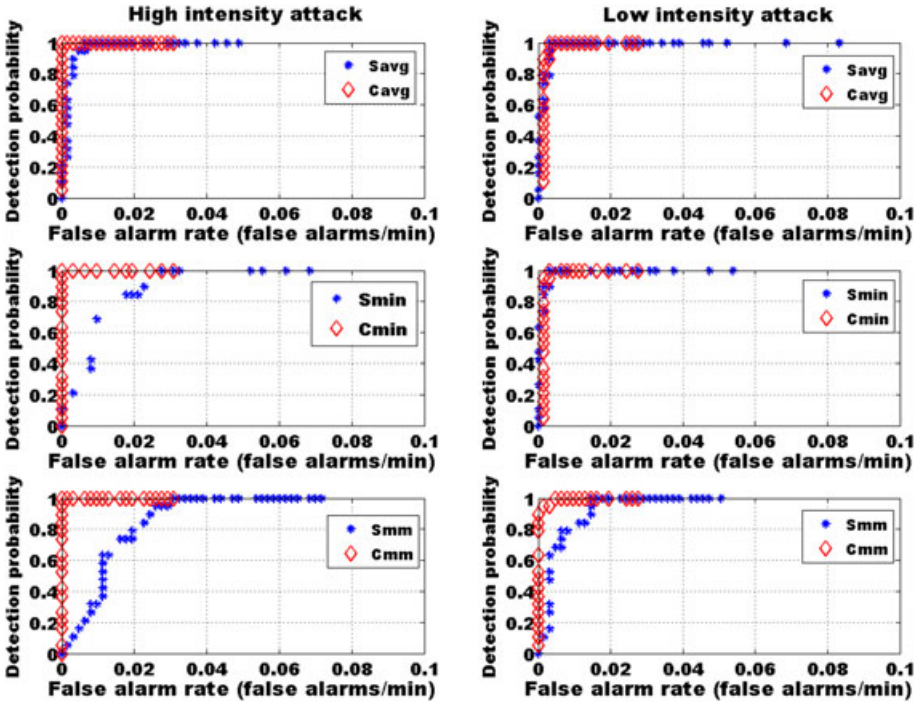


Figure 3. Trade-off points of the local algorithms at Monitor-1 during Exp_1.

select the point that gives the maximum score; that is, the maximum score characterizes the performance of the algorithm. The higher S is, the better the performance of the algorithm. We have also defined that a *DP–FAR* trade-off point is (relatively) robust if its detection threshold needs to change by more than 20%, to change its score S by more than 20%. For the evaluation, described in the next section, we consider only the robust trade-off points.

7. PARAMETER SETTINGS FOR THE LOCAL DETECTION ALGORITHMS

As the equations in Section 5 show, the simple detection algorithms have a single parameter for setting: the detection threshold h. The cusum algorithms also require h, as well as an extra parameter a that controls the drift of expectation Z_n (4). We name a as the *drift coefficient*.

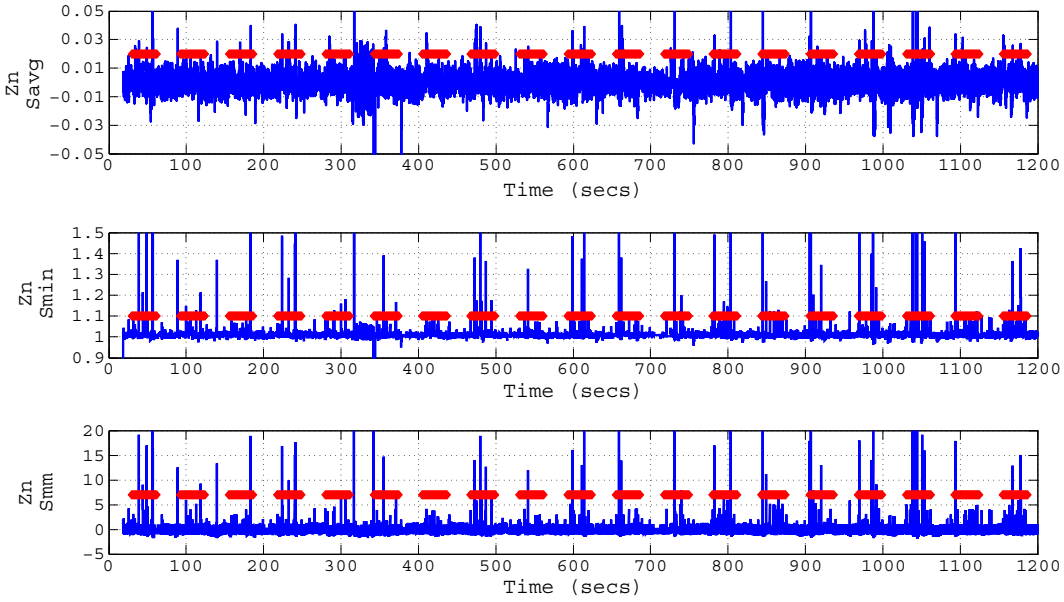


Figure 4. Expectation for the different signal-to-interference-plus-noise ratio-based metrics.

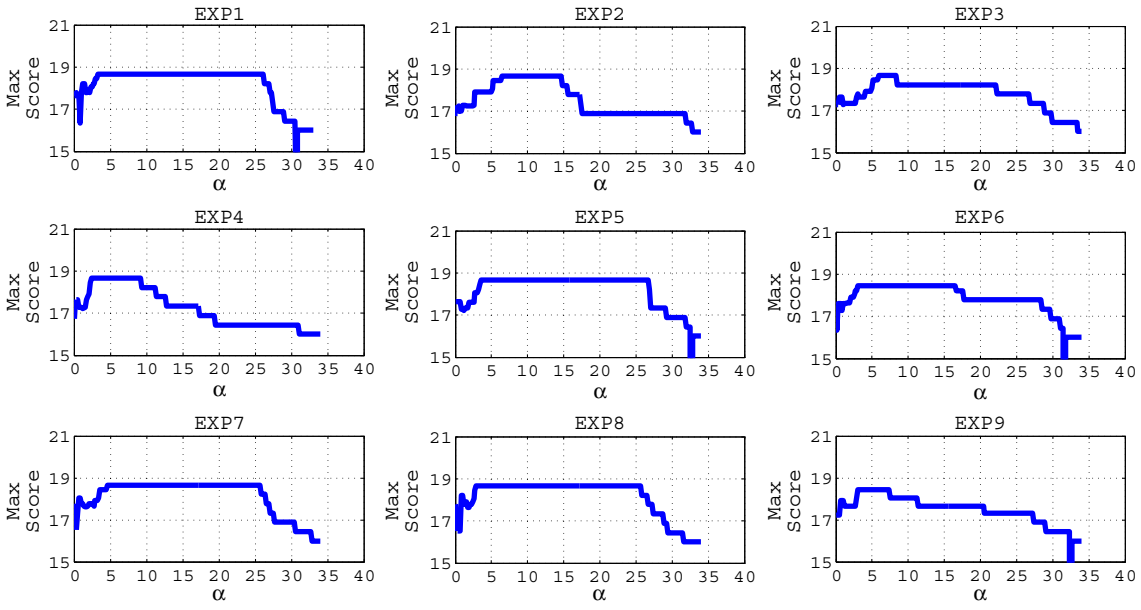


Figure 5. Maximum score versus the drift coefficient for the C_{mm} algorithm.

Note that Z_n is different for each type of algorithm. As an example, Figure 4 shows how the expectation changes during the jamming attacks (shown with the orthogonal boxes) for the different metrics considered.

To find the optimum value of a, we ran a number of experiments (varying the density of the attack), executing our evaluation method jointly considering robustness. Figure 5 shows the maximum scores achieved (we consider only the maximum score that the robust *DP–FAR* points give) for the C_{mm} algorithm and for different values of a. Using this graph, we choose $a = 8$ for this algorithm. For the rest of the cusum algorithms, we produce similar graphs that guide us to select the optimum values of the drift coefficient, allowing us to continue with the performance evaluation in the next sections.

8. PERFORMANCE EVALUATION OF THE LOCAL DETECTION ALGORITHMS

We begin by discussing the performance of the local detection algorithms using the SINR measurements collected at the Receiver (*RCV*), Monitor-1 (MON_1), and Monitor-2 (MON_2).

In general, and depending on the score, a system operator can characterize the performance of an algorithm as low, medium, and so on. For the algorithms evaluated in this work, we characterize the performance as (i) *low* when the score is less than 15, (ii) *medium* when the score is larger than 15 but smaller than 17, (iii) *high* when the score is larger than 17 but smaller than 20, and (iv) *maximum* when the score equals 20. Nevertheless, our technique allows a system operator to define its desired score scale (we use values from 0 to 20) and its own performance characteristics based on the scores (low, medium, high, and so on).

We note here that the long and short windows of the algorithms have been set to 1000 and 10, respectively. Figure 6(a) shows the scores assigned to the algorithms using the SINR traces collected during Exp_1 and Exp_2, and for the high-intensity attack. Recall that Jammer is located at different positions in Exp_1 and Exp_2 (Figure 1). For Exp_1, the evaluation shows that all algorithms at *RCV*, except C_{avg}, achieve maximum performance; thus, they detect all attacks with zero false alarms. At MON_1, all cusums achieve maximum performance, whereas S_{avg} has high performance and S_{min} with S_{mm} has low performance. At MON_2, C_{min} and C_{mm} achieve maximum performance, whereas C_{avg} has high performance. Regarding the simple algorithms, S_{min} achieves high performance, whereas S_{avg} and S_{mm} low.

For Exp_2, and for the high-intensity attack, the performance of all algorithms deteriorates. As mentioned in Section 4, all experiments ran in a public place where people freely walked but with the difference that Exp_1 was conducted late in the evening when most people had left work, whereas Exp_2 early in the noon during busy working hours. SINR is highly affected by many factors such as obstacles and movements of people. We observed that there are SINR drops during Exp_2 at periods when Jammer was inactive. Figure 7 shows the variations of SINR in all monitors during the high-intensity attack of Exp_2 (the jamming attacks are depicted as orthogonal boxes). The gray arrows, on the right side of this figure, show that all monitors recorded SINR variations during times when there were no jamming attacks; possibly, this was the result of people movements. As we verified during the evaluation process, these SINR drops generated a number of false alarms, resulting in performance deterioration that is finally depicted on the right part of Figure 6(a). However, despite these large SINR variations, C_{mm} still achieves high performance at all monitors. Regarding the evaluation at *RCV*, S_{avg} and S_{mm} have low performance, C_{avg} achieves medium performance, whereas the rest of the algorithms achieve high. At MON_1, S_{min} and C_{min} achieve high performance, whereas S_{avg}, C_{avg}, and S_{mm} have low. At MON_2, we observe similar performance as at MON_1.

Figure 6(b) shows the scores assigned to the algorithms using the SINR traces during Exp_1 and Exp_2, and for the low-intensity attack. Regarding Exp_1 and at *RCV*, C_{mm} achieves maximum performance, thus detecting all attacks with no false alarms. The performance of the rest of the algorithms is high. At MON_1, all cusums achieve high performance. Regarding the simple algorithms, only S_{avg} has high performance, whereas the rest have low. At MON_2, we observe similar performance as at MON_1.

For Exp_2 and for the low-intensity attack, the cusums achieve high performance at all monitors. All simple algorithms at all monitors have low performance, except S_{avg} that has medium performance at *RCV* and low at the rest of the monitors.

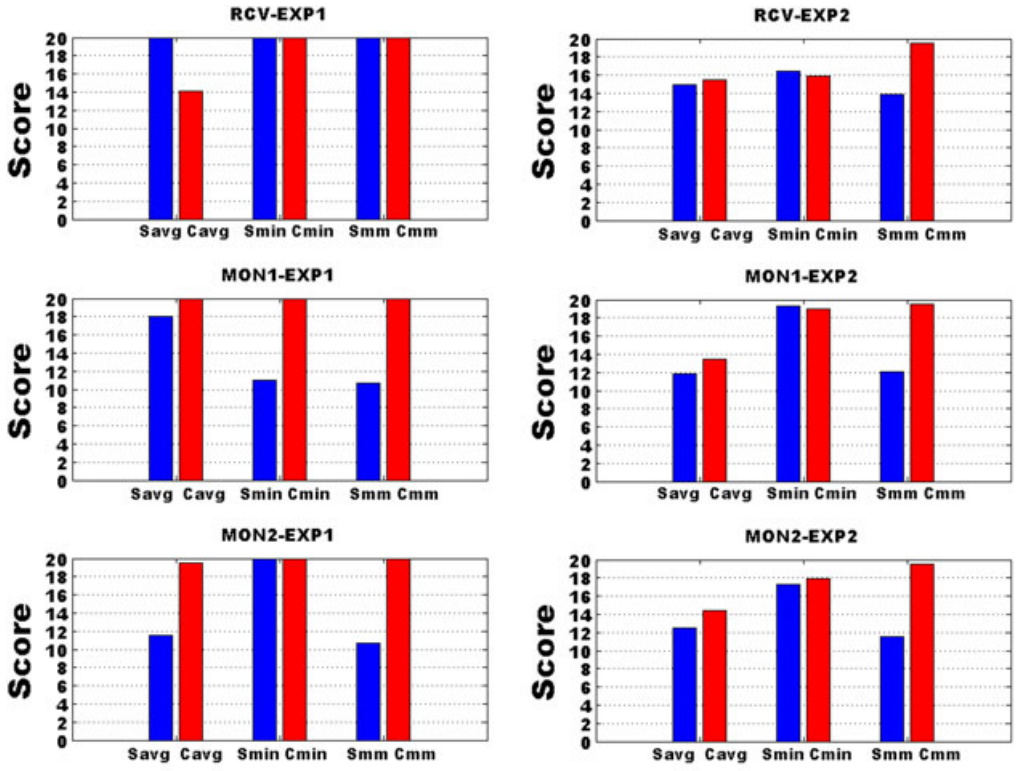
9. COLLABORATIVE INTRUSION DETECTION

In the previous sections, we described the local detection algorithms that execute at each monitor, independently. The evaluation shows that performance is substantially affected by the distance between each monitor and the Jammer, as well as by the type of the algorithm used. Overall, cusum algorithms have higher scores than the simple ones.

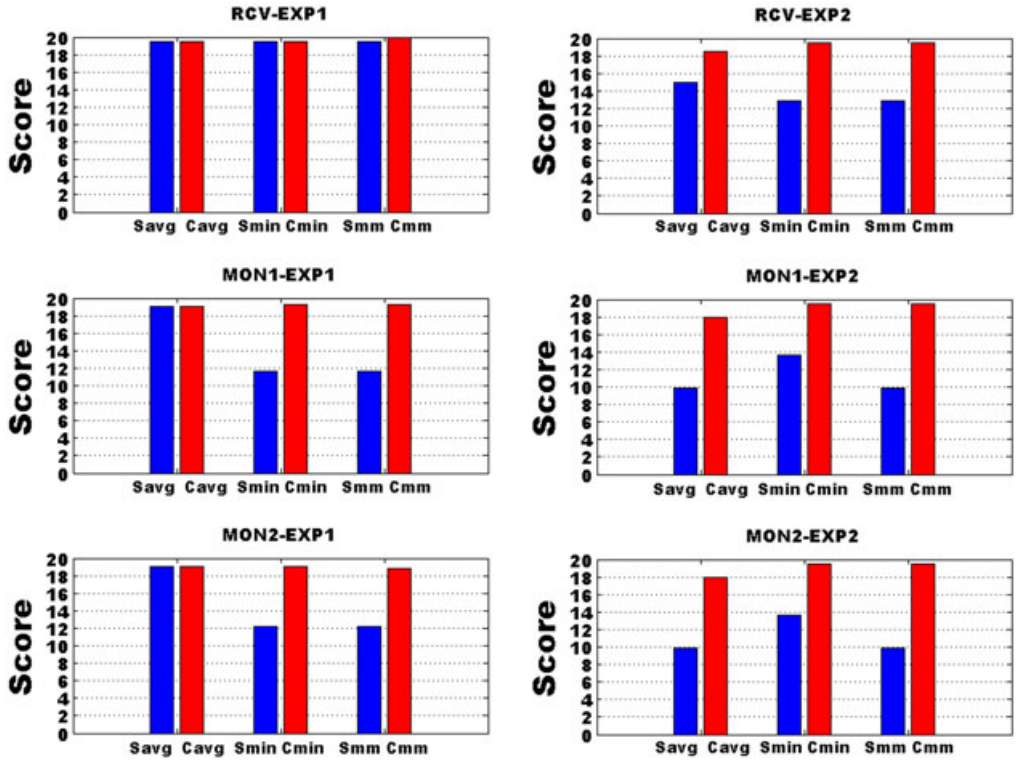
Here, we present the evaluation of a collaborative scheme by using the DS, which combines the outputs of the local algorithms. DS executes in a main fusion node (MFN), an entity with the role to collect and fuse the information provided by the monitors, taking the final decision regarding a possible attack.

Averaging and fusion is applied at three levels as Figure 8 shows:

- *Level 1*, where data are averaged in each monitor at fixed time intervals of duration T_{m}, prior to transmission to MFN. SINR measurements are recorded in a per-packet basis; therefore, the output of the local detection algorithms is proportional to the input



(a) High intensity attack



(b) Low intensity attack

Figure 6. Performance evaluation of the local detection algorithms at all monitors for Exp_1 and Exp_2.

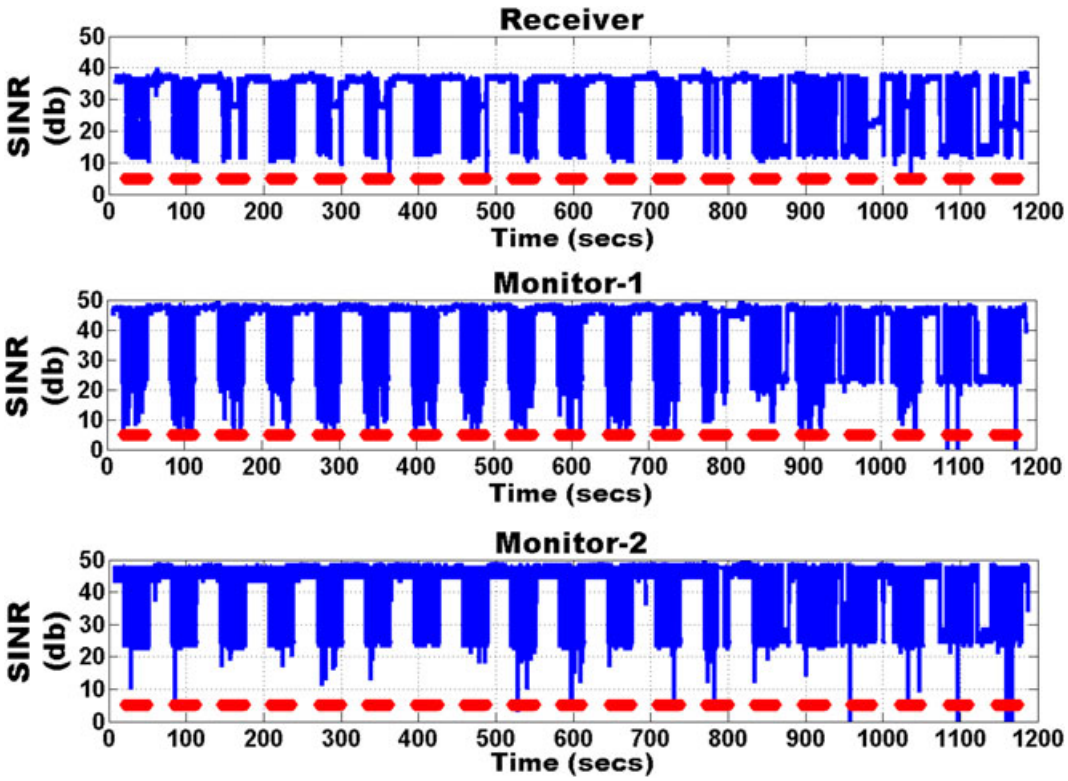


Figure 7. Signal-to-interference-plus-noise ratio (SINR) variations during the high-intensity attack of Exp_2.

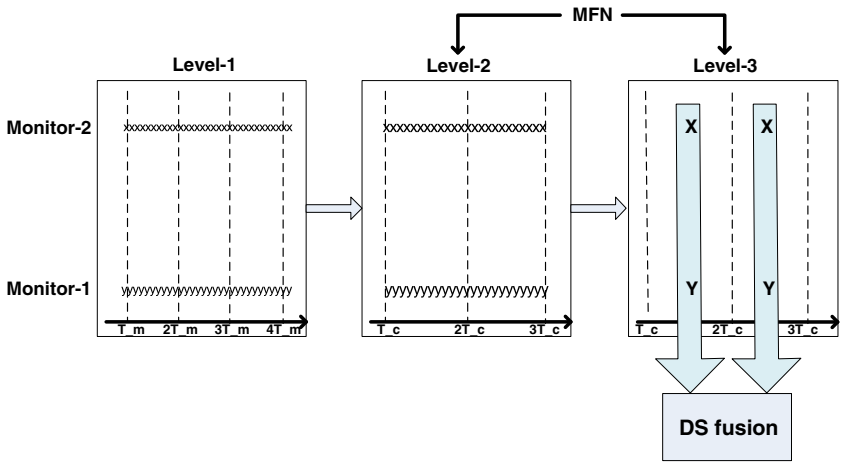


Figure 8. Levels at where data averaging and fusion are applied. MFN, main fusion node; DS, Dempster–Shafer theory of evidence.

packet rate. Averaging at this level saves valuable resources in case of a high-input packet rate. Further work can include the use of a dynamic averaging mechanism that adapts T_m depending on the input packet rate.

- *Level 2*, where the data sent by each monitor are averaged by MFN at fixed time intervals of duration T_c. Averaging is applied on the data of each monitor independently from the data of the rest of the monitors.
- *Level 3*, the final level where the data of the different monitors are fused together, to produce the final output. This is the level where DS is used.

Figure 8 shows the different averaging and fusion levels for a collaborative intrusion detection system (CIDS) of two monitors. X and Y symbolize the outputs of Monitor-1 and Monitor-2, respectively. This scheme can be extended for any number of monitors. In this figure, observe that at

Level 3, there is only a single output per monitor during a time interval of duration T_c. The fusion rule applied at Level 3 fuses the outputs of the different monitors together, whereas at Levels 1 and 2, averaging is applied on the outputs of each monitor separately. We note here that fusion and averaging are performed offline in Matlab code that uses the real traces collected using the testbed of Figure 1.

Both T_m and T_c are operator controlled. A very small value of T_m will increase the amount of data sent to MFN, and hence, it will waste a large amount of wireless resources. On the other hand, a very large value of T_m may lead to a large number of undetected attacks. Also, a very large value of T_c will require less computational resources in MFN, but it may lead to a large number of undetected attacks.

9.1. Dempster–Shafer theory of evidence

The basic DS was first introduced in [8] as a mathematical framework for the representation of uncertainty. The main advantage of this algorithm is that no a priori knowledge of the system is required, thus making it suitable for anomaly detection of previously unseen information [20]. We exploit this advantage as our anomaly detection algorithms are based on the SINR. As it is well known, SINR is volatile; thus, no models exist to describe its fluctuations under different network conditions. Another advantage of DS is its usefulness in combining data sent by different observers (monitors) [22].

Essential terminologies related to DS and used by this work are as follows:

- Frame of discernment (Θ). This is the set of all possible mutually exclusive and complete states of a system $\Theta = \{\theta_i | 1 \leqslant i \leqslant N\}$ [22]. For an intrusion detection system, two mutually exclusive and complete states that can be defined are θ_1:attack and θ_2:normal, so the frame of discernment is $\Theta = \{\theta_1, \theta_2\}$ or equivalently, $\Theta = \{attack, normal\}$.
- Probability assignment function (or mass function). This function is a primitive of theory of evidence. It is usually symbolized by m. As defined in [8], if Θ is a frame of discernment, then function $m : 2^\theta \rightarrow [0, 1]$ is called a basic probability assignment whenever

$$m(\phi) = 0$$

and

$$\sum_{A \subset \Theta} m(A) = 1.$$

The mass value of A ($m(A)$) is also called A's basic probability number, and it is understood to be the measure of the belief that is committed exactly to A. A is a subset of Θ, and its mass function supports a belief on the basis of some evidence. In our intrusion detection system, this evidence is based on the SINR values.

- Belief function. This function measures the belief of a proposition A, and it computes the sum of all the nonempty subsets of A. It is given by the following formula:

$$Bel(A) = \sum_{B \subseteq A} m(B) \tag{6}$$

For a cross-layer intrusion detection system, proposition A could be the hypothesis of *attack*, and the subsets B of A could be defined as B_1:*jamming attack*, B_2:*ICMP flooding attack*, B_3:*SYN flooding attack*, and so on; therefore, the belief function of A requires evidence for all of its subsets.

- Focal elements. The focal elements of a frame of discernment Θ consists of all hypotheses; observers (i.e., monitors) can provide evidence (or express beliefs). If for example $\Theta = \{attack, normal\}$, then the focal elements of this frame of discernment are [attack, normal, (attack or normal)]; therefore, the monitors of a collaborative intrusion detection system can send beliefs to MFN regarding these three focal elements. This is the type of focal elements we use in this work.

DS has the ability to combine evidence from different information sources. Let us assume that there are two information sources, in our case two monitors, and then supposing Monitor-1 believes that hypothesis A is true with confidence $m_1(A)$ and Monitor-2 believes that hypothesis A is true with confidence $m_2(A)$, DS combines these two separate beliefs into a single belief:

$$m_{12}(A) = \frac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1 - K} \tag{7}$$

where

$$K = \sum_{B \cap C = \varnothing} m_1(B) m_2(C) \tag{8}$$

The fusion rule in (7) is characterized by iteration [22]; thus, it can be treated as a single source that can be further combined with evidence provided by the third source. If we symbolize the fusion on the basis of DS as $\bigoplus$, then (7) can be written as $m_{12}(A) = m_1(B) \bigoplus m_2(C)$. If there are three sources, the combined evidence is $m_{123}(A) = m_1(B) \bigoplus m_2(C) \bigoplus m_3(D)$ or $m_{123}(A) = m_{12}(B) \bigoplus m_3(D)$ or equivalently, $m_{123}(A) = m_1(B) \bigoplus m_{23}(C)$, denoting the associative characteristic of DS.

K in (7) represents a basic probability mass related to conflict. Conflict can appear when, for example, Monitor-1 express a strong belief about hypothesis θ_1:*attack* and Monitor-2 a strong belief about hypothesis θ_2:*no attack*. The denominator of (7) normalizes the combined belief $m_{12}(A)$; thus, it attributes any

Table I. Belief computation for the Dempster–Shafer algorithm.

Belief/y_n	$y_n < h(1-r)$	$y_n \geq h(1+r)$	$h(1-r) \leq y_n \leq h(1+r)$
b_a	$\frac{1-b_n}{2}$	$c + \frac{y_n - h}{h}$	0.333
b_n	$c + \frac{h-y_n}{h}$	$\frac{1-b_a}{2}$	0.333
b_{na}	$\frac{1-b_n}{2}$	$\frac{1-b_a}{2}$	0.333

mass function associated with conflict to the null mass [38]. An alarm is raised if $m_{123}(A) > q$, where $q > 0$ is operator controlled.

DS combines the beliefs expressed by the three monitors (*RCV*, MON_1, and MON_2) producing a single combined belief that is finally compared with $q = 0.5$. If the combined belief is greater than q, an alarm is raised. The monitors (based on the local detection algorithms) produce a single belief for each focal element: (i) b_a the belief that there is an attack, (ii) b_n the belief that there is not an attack (normal), and (iii) b_{na} the belief expressing an ambiguity: attack or no attack. If the output of a local detection algorithm is close to h, where h is the detection threshold, b_{na} increases to express a higher belief on the uncertainty of an attack or normal operation.

The region R where uncertainty increases is operator controlled and is defined between the borders of $h \times (1-r)$ and $h \times (1+r)$, as $h \times (1-r) \leq R \leq h \times (1+r)$ and $r > 0$. The width of the uncertainty area is controlled by r. By using this definition, beliefs are computed according to Table I, depending on the output y_n of the local detection algorithms. Beliefs are computed so as their sum is equal to one ($b_a + b_n + b_{na} = 1$). An alarm is raised if $b_c > q$, where b_c is the combined belief given by (7) and q the predefined threshold used for fusion. As we focus only on detecting a single type of attack (jamming at the physical layer), the belief and mass functions are equivalent. For the performance evaluation, we have selected $r = 0.05$ and $c = 0.5$. c is a constant that controls how fast b_a or b_n increase (Table I). In Table I, $y_n = Z_n$ for the simple algorithms, and $y_n = \max(0, y_{n-1} + Z_n - a)$ for the cusum ones (as described in Section 5).

DS has been criticized that it performs poorly when there is significant conflict among the different information sources fused. However, as the evaluation shows (Section 9.2), DS significantly increases the performance of the local algorithms even if there are possible conflicts among the different monitors. Moreover, a major objective of this work is to investigate if fusion can increase the performance of the local intrusion detection algorithms. Further work can include the study of other fusion algorithms with conflict solving.

9.2. Performance evaluation of the Dempster–Shafer algorithm

In this section we evaluate DS in terms of the *DP* and *FAR*. By evaluating the local detection algorithms (Section 8), we collected sets of detection threshold values for each monitor and for each local algorithm, separately. These threshold sets correspond to robust *DP–FAR* points achieving the highest possible score. We denote as $thr_i = \{thr_{i1}, \ldots, thr_{iN}\}$ the set of the detection thresholds of monitor i for which a local detection algorithm is robust while achieving its highest possible score. As in this work, three monitors are used; there exist three sets of threshold values, one for each monitor: (thr_1, thr_2, thr_3). We next combine these sets to take triplets of all possible combinations.

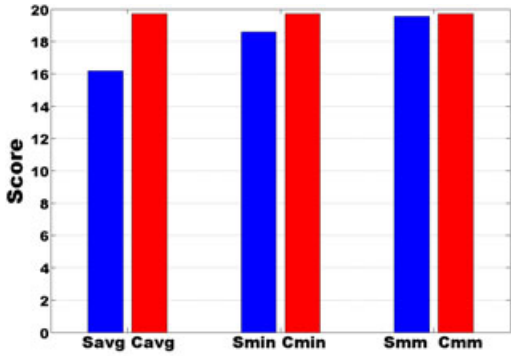
We evaluate DS not by assigning the same detection threshold to all monitors but choosing the threshold values that emerged from the aforementioned combinations (some preliminary tests have shown that DS does not perform well if the same threshold is set to all monitors). For example, instead of assigning threshold thr_1 to all monitors and then use DS to fuse their outputs, we can assign thr_{11} to *RCV*, thr_{21} to MON_1, and thr_{31} to MON_2, as [thr_{11}, thr_{21}, thr_{31}] is one possible combination. There are hundreds of thousands of possible combinations, and it is computationally infeasible to evaluate DS by using all of these values. For this reason, we use only a subset of all the possible combinations. For each algorithm, we uniformly select triplets from the sets [thr_{1j}, thr_{2k}, thr_{3l}], where j, k, l is the number of the robust thresholds that give the highest score at *RCV*, MON_1, and MON_2, respectively.

DS evaluation using the subset of threshold triplets gives several different scores, but finally, the highest score among all of these values is assigned. Figure 9 shows the scores assigned to DS, when combined with the local detection algorithms, and for both attack intensities. Observe in this figure that DS with S_{avg}, for Exp_1 and for both attack intensities, has medium performance. When combined with the rest of the local algorithms, its performance is high for both experiments and for both attack intensities.

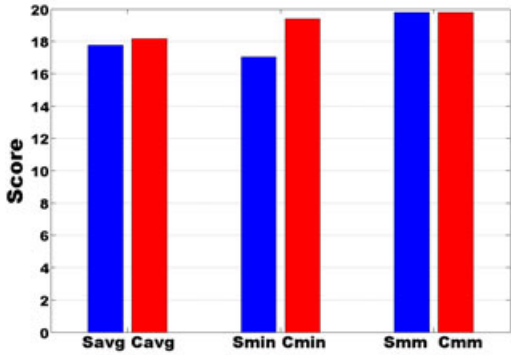
To quantify the performance improvement when using DS, compared with the performance of the local algorithms evaluated exclusively at the single locations (monitors), we define two metrics:

- The performance improvement metric P, which quantifies the performance improvement when using *DS* combined with a local detection algorithm, compared with the performance of this algorithm at a single location (9).

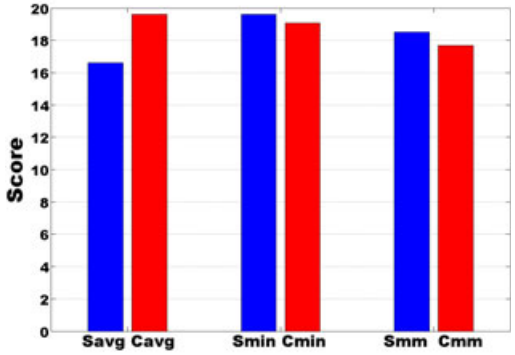
$$P^{i,j} = 100 \times \frac{S^j_{DS} - S^j_i}{S^j_i} \tag{9}$$



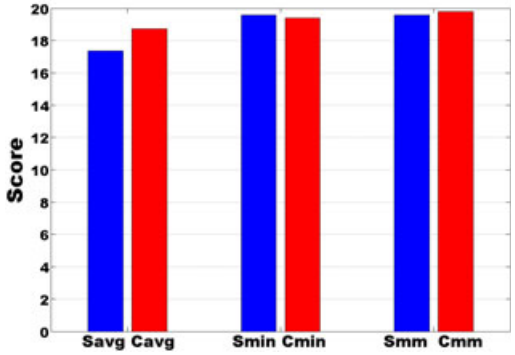
(a) High intensity attack, Exp_1



(b) High intensity attack, Exp_2



(c) Low intensity attack, Exp_1



(d) Low intensity attack, Exp_2

Figure 9. Scores assigned to Dempster–Shafer theory of evidence for the high-intensity and low-intensity attacks.

where S_{DS}^{j} is the score assigned to DS when combined with the local detection algorithm j and S_i^j is the score assigned to the local detection algorithm j when considering the SINR traces at monitor i exclusively ($i \in [RCV, MON_1, MON_2]$ and $j \in [S_{\mathrm{avg}}, S_{\mathrm{min}}, S_{\mathrm{mm}}, C_{\mathrm{avg}}, C_{\mathrm{min}}, C_{\mathrm{mm}}]$).

- The average performance improvement metric P_avg that is the average value of the performance improvement when using DS, given by (10).

$$P\_\mathrm{avg}^{j} = 100 \times \frac{\sum_{i=1}^{N} \frac{S_{\mathrm{DS}}^{j} - S_i^j}{S_i^j}}{N} \tag{10}$$

where N is the number of monitors, S_{DS}^{j} the score assigned to DS when combined with the local algorithm j, and S_i^j is the score assigned to this local algorithm when evaluated exclusively at monitor i.

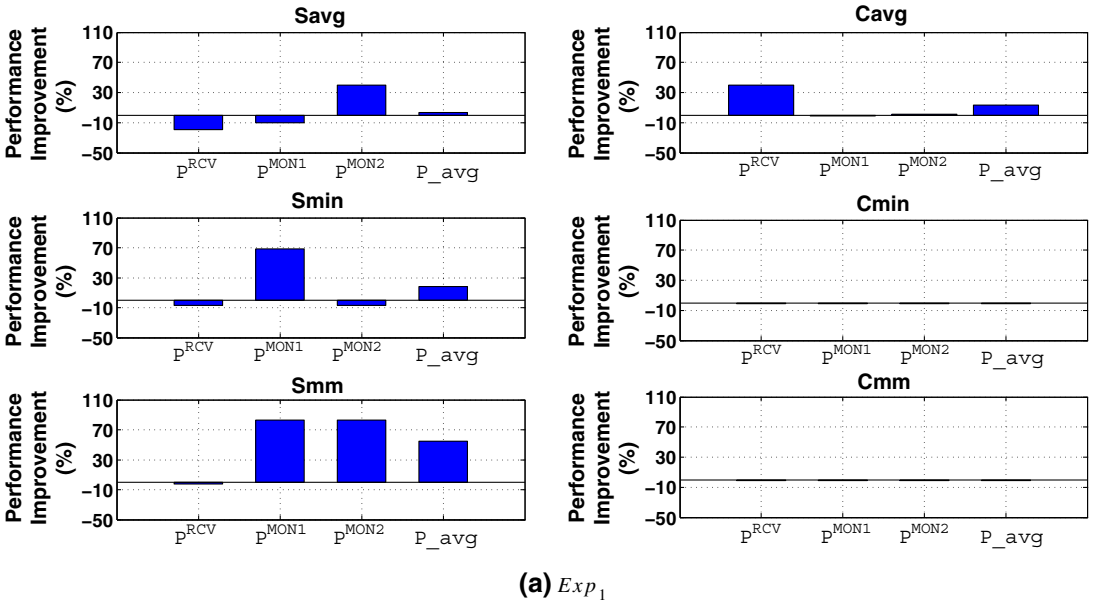
These two performance metrics quantify the comparison between Figure 6(a) and (b) that shows the performance of the local algorithms evaluated at the single locations, with Figure 9 that shows the scores assigned to DS. These metrics eventually assist to investigate the possible performance improvement gained when using DS.

Figures 10 and 11 show the performance improvement (P) and the average performance improvement ($P_$avg) when using DS. For simplicity and for each graph on these figures,

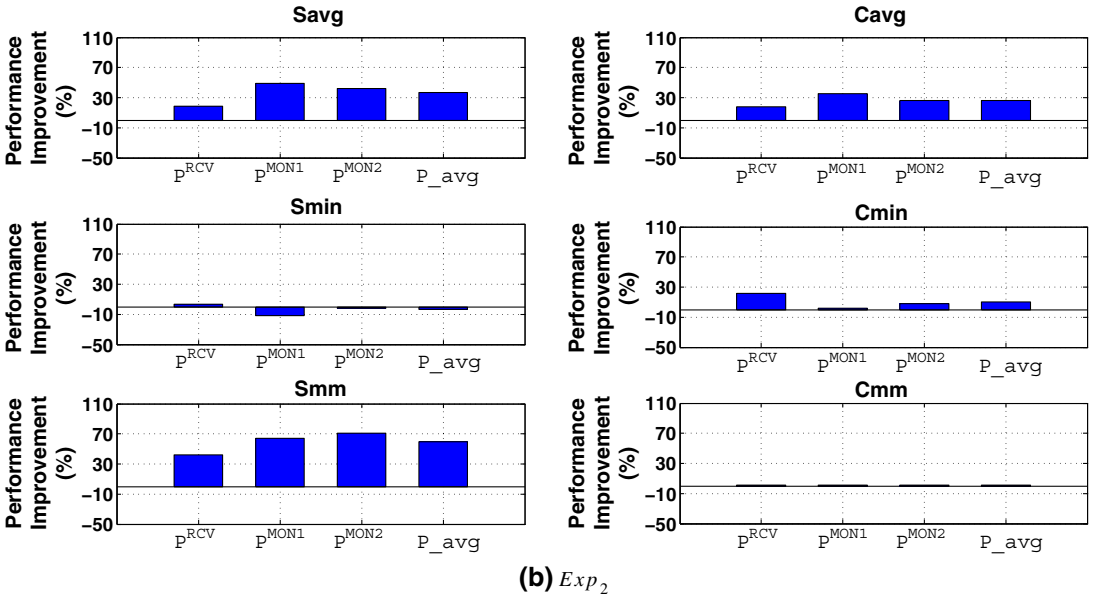
- The first three bars show the metric P^i, where i is the monitor on which the algorithm, shown as the title of the graph, has been exclusively evaluated.
- The fourth bar shows the metric $P_$avg that is the average value denoted by the first three bars.

In Figure 10(a), observe that $P_$avg is greater than zero for the simple algorithms and C_{avg}, meaning that their performance increases if the outputs of the monitors where they execute are fused using DS. In Figure 6(a), we observe that S_{min} and S_{mm} in MON_1 and MON_2, and C_{avg} in RCV have low performance, but when DS is used, their performance increases, achieving high performance. The average performance improvement for S_{min} is 18%, for S_{mm} is 55%, whereas for C_{avg}, the performance improves by 13%. For C_{min} and C_{mm}, there is no performance improvement as both reach maximum performance when evaluated at the single locations. With DS, they still have high performance, although fusion decreases their scores by 1.5%.

Figure 10(b) shows the performance improvement for Exp_2 and for the high-intensity attack. Recall from Figure 6(a) that for Exp_2, all simple algorithms except S_{min} that achieves medium performance have low scores. When DS is used, their performance substantially increases as DS improves the average performance by 37% for S_{avg} (achieving medium performance now) and by 59% for S_{mm} (achieving high performance now). Also for Exp_2,



(a) Exp_1



(b) Exp_2

Figure 10. Performance improvement when using Dempster–Shafer theory of evidence for the high-intensity attack.

C_{avg} has medium or low performance when evaluated at the monitors, but when fusion is used, its average performance improves by 26%, achieving high performance. For the rest of the algorithms ($S_{\min}$, $C_{\min}$, and C_{mm}), DS does not highly affect their performance as they already achieve high performance at all monitors they were exclusively evaluated.

The performance improvement for Exp_1 and the low-intensity attack is shown in Figure 11(a). DS substantially increases the average performance of $S_{\min}$ and S_{mm} by 43% and 35%, respectively. These algorithms have low performance at MON_1 and MON_2 (Figure 6(b)), whereas with DS, both achieve high performance. The rest of the algorithms have high performance at the monitors, and DS does not highly affect their performance (less than $\pm 10\%$ of performance variation).

Finally, the average performance improvement for Exp_2 and for the low-intensity attack is shown in Figure 11(b). DS enhances the performance of the simple algorithms by 56% for S_{avg}, 46% for $S_{\min}$, and 83% for S_{mm}. With fusion, the performance of the simple local algorithms increases from low (Figure 6(b)) to high. The performance of the cusum algorithms is not highly affected by fusion (less than 5%), and it still remains high.

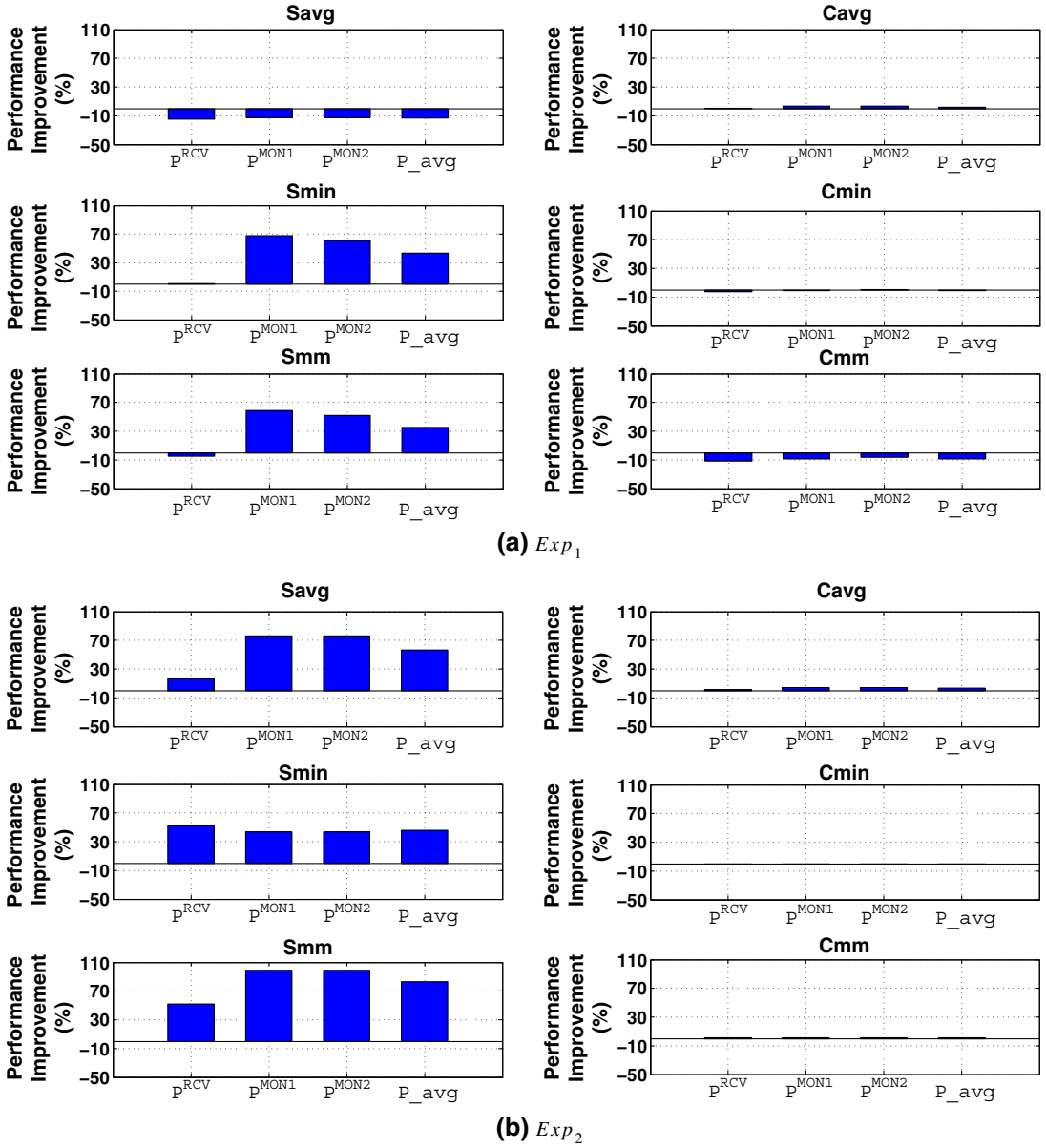


Figure 11. Performance improvement when using Dempster–Shafer theory of evidence for the low-intensity attack.

10. CONCLUSIONS AND FURTHER WORK

In this work, we described and evaluated anomaly-based intrusion detection algorithms for the detection of jamming attacks at the physical layer of a wireless network. The algorithms execute locally at each monitor seeking for changes in the statistical characteristics of SINR, and they are of two types: simple threshold and cusum-type algorithms. We collected SINR traces from three locations (RCV, MON_1, MON_2) of a real IEEE 802.11 experimental network and evaluated the algorithms in terms of the detection probability (DP), false alarm rate (FAR), and their robustness to different detection threshold values, under two attack intensities. We conducted two experiments: Exp_1 that ran late in the evening when most people had left work and Exp_2 that ran during busy working times when people could freely walked.

For the evaluation of the algorithms, we used a method that, on the basis of the produced outputs of each monitor, filters the robust thresholds, and it then assigns a score for these thresholds on the basis of DP and FAR. Opposed to other similar contributions, we also considered the robustness of the algorithms for different detection thresholds, hence providing a more comprehensive performance evaluation.

For Exp_1, and when the measurements at *RCV* are used, for both attack intensities, all algorithms (except C_{avg}) achieve maximum performance (all attacks detected with zero false alarms) for detection threshold values they are robust (*RCV* is located very close to Jammer). When the measurements at MON_1 and MON_2, which are located in a larger distance from Jammer, are considered, and for both attack intensities, cusums achieve maximum or high performance. Simple algorithms' performance varies from low to high depending on the monitor considered.

For Exp_2, and for the high-intensity attack, the performance of most algorithms deteriorates. This is because of the SINR variations during Jammer inactivity. These variations were recorded by all monitors and were probably caused by peoples' movements, as Exp_2 was conducted during busy working hours. Nevertheless, C_{mm} still achieves high performance. The performance of the rest of the algorithms varies from low to high, depending on the monitor considered.

For the low-intensity attack of Exp_2, all cusum algorithms achieve high performance. Again, the performance of the simple algorithms varies depending on the monitor considered.

The evaluation of the local detection algorithms shows that, in general, cusum algorithms achieve high performance for both experiments and for both attack intensities. The performance of the simple algorithms can vary from low to high, depending on the algorithm and the monitor used. Among all algorithms, C_{mm} achieves high or maximum performance, regardless the attack intensity and the Jammer location.

Next, we presented the DS. Its main advantage is that no a priori knowledge of the system is required, thus making it suitable for anomaly detection of previously unseen information. DS is used to fuse the outputs provided by the local detection algorithms that executed at the monitors. We used a subset of the robust detection thresholds that were derived through the evaluation of the local algorithms, and then, we evaluated DS in terms of *DP* and *FAR*.

The evaluation shows that when DS is combined with all the local detection algorithms (except S_{avg}), it has high performance. The performance of the simple algorithms substantially increases with an average performance improvement (in some cases) more than 80% when their outputs are fused by DS. Regarding S_{avg}, it achieves medium performance for the high-intensity attack of both experiments with an average performance improvement of 40%. The average performance improvement for the cusum algorithms is not high, as they already achieve high scores even when they are evaluated at the monitors exclusively.

Further work can include the launching and detection of more sophisticated jamming attacks. Such an attack could be performed by a Jammer through the use of a directional antenna that targets an access point. The use of a directional antenna makes more difficult the detection of jamming because the noise radiated is concentrated in a sector possibly located outside the detection range of the monitors. In this situation, measurements from the MAC layer such as the retry counter or the CCA counter could be used for detection.

Other types of attacks can be studied targeting higher-layer protocols (i.e., IP and transport), as wireless networks are also susceptible to attacks targeting traditional wired network infrastructures. A cross-layer intrusion detection system composed of algorithms and mechanisms monitoring activities in different layers is under consideration. A single combined output (or verdict) can be produced by fusing the output of each distinct layer-level mechanism. One of the algorithms for data fusion of heterogenous information sources is DS and several of its variations for conflict solving.

Furthermore, we aim to investigate appropriate training schemes so as to select the most optimal values for the tuning parameter a, making the cusum algorithms more robust when deployed in unfamiliar networks.

ACKNOWLEDGEMENT

This work was supported in part by the European Commission through FP7 Project EU-MESH, ICT-215320.

REFERENCES

1. Pelechrinis K, Iliofotou M, Krishnamurthy S. Denial of service attacks in wireless networks: the case of jammers. *IEEE Communications Surveys & Tutorials* 2011: 245–257.
2. Bicacki K, Tavli B. Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks. *Computer Standards and Interfaces (Elsevier)* September 2008; **2009**: 931–941.
3. Thuente D, Newlin B, Acharya M. Jamming vulnerabilities of IEEE 802.11e. In *Proceedings of MILCOM 2007, IEEE*, Orlando, USA, October 2007; 1–7.
4. Xu W, Ma K, Trappe W, Zhang Y. Jamming sensor networks: attack and defense strategies. *IEEE Network* May 2006; **20**: 41–47.
5. Hall M, Silvennoinen A, Haggman S. Effect of pulse jamming on IEEE 802.11 wireless LAN performance. In *Proceedings of MILCOM 2005, IEEE*, Atlantic City, USA, October 2005; 2301–2306.
6. Cardenas A, Radosavac S, Baras J. Evaluation of detection algorithms for mac layer misbehavior: theory and experiments. *IEEE/ACM Transactions on Networking* 2009; **17**: 605–617.
7. Fragkiadakis A, Tragos E, Askoxylakis I. Design and performance evaluation of a lightweight wireless early warning intrusion detection prototype. *EURASIP Journal on Wireless Communications and Networking* 2012; **12**: 1–18.

8. Shafer G. *A Mathematical Theory of Evidence*. Princeton University Press: Princeton, New Jersey, USA, 1976.
9. Sheth A, Doerr C, Grunwald D, Han R, Sicker D. MOJO: a distributed physical layer anomaly detection system for 802.11 WLANs. In *ACM MobiSys*, Upsala, Sweden, 2006; 191–204.
10. Aime M, Callandriello G, Lioy A. A wireless distributed intrusion detection system and a new attack model. In *IEEE Symposium on Computers and Communications*, Cagliari, Italy, 2006; 35–40.
11. Tomko A, Rieser C, Buell L. Physical layer intrusion detection in wireless networks. In *Military Communications Conference*, 2006; 1–7.
12. Wood A, Stankovic J, Zhou G. Deejam: defeating energy-efficient jamming in ieee 802.15.4-based wireless networks. In *Proceedings of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, San Diego, USA, June 2007; 60–69.
13. Siris V, Papagalou F. Application of anomaly detection algorithms for detecting syn flooding attacks. *Computer Communications* 2006; **29**(9): 1433–1442.
14. Cabrera J, Gutierez C, Mehra R. Ensemble methods for anomaly detection and distributed intrusion detection in mobile ad-hoc networks. *Information Fusion (Elsevier)* 2008; **9**(1): 96–119.
15. Peng T, Leckie C, Ramamohanarao K. Information sharing for distributed intrusion detection systems. *Journal Of Network And Computer Applications (Elsevier)* 2007; **30**(3): 877–899.
16. Chen Y, Hwang K, Ku W. Distributed change-point detection of DDoS attacks: experimental results on deter testbed. In *Proceedings of USENIX Security Symposium*, Boston, USA, August 2007; 1–7.
17. Chatzigiannakis V, Androulidakis G, Pelechrinis K, Papavassiliou S, Maglaris V. Data fusion algorithms for network anomaly detection: classification and evaluation. In *ICNS '07: Proceedings of the Third International Conference on Networking and Services*. IEEE Computer Society: Washington, DC, USA, 2007; 1–7.
18. Aparicio F, Kyriakopoulos K, Parish D. An on-line wireless attack detection system using multi-layer data fusion, In *IEEE International Workshop on Measurements and Networking Proceedings*, Anacapri, Italy, 2011; 1–5.
19. Siaterlis C, Maglaris B. Towards multisensor data fusion for DoS detection. In *SAC '04: Proceedings of the 2004 ACM Symposium on Applied Computing*. ACM: New York, NY, USA, 2004; 439–446.
20. Chen Q, Aickelin U. Anomaly detection using the Dempster-Shafer method, In *Proceedings of the 2006 International Conference on Data Mining, DMIN 2006*, Las Vegas, USA, 2006; 232–240.
21. Li W, Joshi A. Outlier detection in ad hoc networks using Dempster-Shafer theory. In *MDM '09: Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*. IEEE Computer Society: Washington, DC, USA, 2009; 112–121.
22. Yu D, Frincke D. Alert confidence fusion in intrusion detection systems with extended Dempster-Shafer theory. In *ACM-SE 43: Proceedings of the 43rd Annual Southeast Regional Conference*. ACM: New York, NY, USA, 2005; 142–147.
23. Cakiroglou M, Ozcerit T. Jamming detection mechanisms for wireless sensor networks. In *Proceedings of 3rd International Conference on Scalable Information Systems*, Napoli, Italy, June 2008; 1–8.
24. Bhuse V, Gupta A. Anomaly intrusion detection in wireless sensor networks. *Journal of High Speed Networks* 2006; **15**: 33–51.
25. Xu W, Trappe W, Zhang Y, Wood T. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of ACM MobiHoc*, Urbana, USA, May 2005; 46–57.
26. Thamilarasu M, Mishra S, Sridhar R. A cross-layer approach to detect jamming attacks in wireless ad hoc networks. In *Proceedings of MILCOM 2006*, Washington DC, USA, October 2006; 1–7.
27. Pelechrinis K, Broustis I, Krishnamurthy S, Gkantsidis C. Ares: an anti-jamming reinforcement system for 802.11 networks. In *Proceedings of CoNext 2009*, Rome, Italy, 2009; 181–192.
28. Linux wireless drivers, ath5k. [Online]. Available: http://linuxwireless.org/en/users/Drivers/ath5k.
29. NTP: the network time protocol. [Online]. Available: http://www.ntp.org.
30. Sampath A, Dai H, Zheng H, Zhao B. Multi-channel jamming attacks using cognitive radios. In *Proceedings of ICCCN 2007*, Honolulu, USA, 2007; 352–357.
31. Airmagnet wifi analyser. [Online]. Available: http://www.airmagnet.com/.
32. Nadgir M, Premkumar K, Kumar A, Kuri J. Cusum based distributed detection in wsns. In *MCDES 2008*, Bangalore, India, 2008; 1–8.
33. Yan G, Xiao Z, Eidenbenz S. Catching instant messaging worms with change-point detection techniques, In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, San Francisco, USA, April 2008; 1–10.
34. Verdier G, Hilgert N, Vila J. Adaptive threshold computation for cusum-type procedures in change detection and isolation problems. *Elsevier, Computational Statistics and Data Analysis* 2008; **52**: 4161–4174.

35. Lu K, Wu D, Fan J, Todorovic S, Nucci A. Robust and efficient detection of DDoS attacks for large-scale internet. *Computer Networks* 2007; **51**(18): 5036–5056.
36. Chen Y, Hwang K, Ku W-S. Collaborative detection of DDoS attacks over multiple network domains. *IEEE Transactions Parallel Distributed Systems* 2007; **18**(12): 1649–1662.
37. Sheng Y, Tan K, Chen G, Kotz D, Campbell A. Detecting 802.11 MAC layer spoofing using received signal strength. In *Proceedings of INFOCOM'08*. IEEE: Phoenix, USA, 2008; 1768–1776.
38. Yager R. On the Dempster-Shafer framework and new combination rules. *Information Sciences* 1987; **41**: 93–137.

AUTHORS' BIOGRAPHIES

Alexandros G. Fragkiadakis is a research associate in the Institute of Computer Science of the Foundation for Research and Technology, Hellas (FORTH-ICS). He received his PhD in Computer Networks from the Department of Electronic and Electrical Engineering of Loughborough University, UK. He has also received his MSc in Digital Communications Systems, awarded with distinction, from the same university. He obtained his diploma degree in Electronic Engineering from the Technological Educational Institute of Piraeus, Greece. He has worked as a research associate within the High Speed Networks Group of the Department of Electronic and Electrical Engineering in Loughborough University. Within FORTH-ICS, he has been involved in several projects in the area of wireless communications and networking. His research interests include wireless networks, intrusion detection and security in wireless networks, reprogrammable devices, open source architectures, cognitive radio networks, and wireless sensor networks.

Vasilios A. Siris received his BS (1990) degree in Physics from the University of Athens, Greece, his MS (1992) in Computer Science from Northeastern University, Boston, USA, and his PhD (1998) in Computer Science from the University of Crete, Heraklion, Greece. Since February 2009, he is an assistant professor at the Department of Informatics of the Athens University of Economics and Business. Prior to that, he was an assistant professor at the Department of Computer Science of the University of Crete, from 2002 to 2009. He is also a research associate at FORTH-ICS, where he was a research assistant and then a researcher from 1993 to 2002. During the summer of 2001, he was a research fellow at British Telecommunication's Research Labs at Adastral Park, Ipswich, UK, where he worked on resource and congestion control for 3G wireless networks. He was the project coordinator of the EU-MESH project. His current research interests include measurement and analysis of network traffic, seamless resource control in wireless and wired networks and flexible charging schemes based on resource usage for service level agreements. He is a member of the IEEE Communication Society.

Nikolaos E. Petroulakis is a research scientist in the Telecommunications and Networks Laboratory of the Institute of Computer Science at FORTH since 2007. He received his degree in Mathematics (2004) from the National and Kapodistrian University of Athens, Greece, and his MSc in Digital Communications (2006) from the Department of Engineering and Design of the University of Sussex, UK. He is also a part-time member of the Computer Emergency Response Team of FORTH (FORTHcert) since 2009. He has been actively involved in several European and National research projects. His research interests include wireless networks, cognitive radios, wireless sensor networks, and network security. He is a member of the IEEE Communication Society.

Apostolos P. Traganitis joined FORTH-ICS in 1988, and since then, he coordinated and participated in a number of EU and nationally funded projects in the Wireless Communications and Health Care sector. He is also a professor in the Department of Computer Science of the University of Crete, where he teaches and does research in the areas of digital communications, wireless networks, communications security, and hardware design and biomedical engineering. During 1993 and 1994, he was a visiting research fellow at the Center of Satellite and Hybrid Communications Networks of the Institute of Systems Research, University of Maryland, USA. Previously, he has been a researcher in the Hellenic Navy Research Laboratory (GETEN), in charge of the Electronic Warfare Unit.