Anomaly-Based Intrusion Detection Algorithms for Wireless Networks*

Alexandros G. Fragkiadakis**, Vasilios A. Siris***, and Nikolaos Petroulakis

Institute of Computer Science, Foundation for Research and Technology - Hellas (FORTH)
P.O. Box 1385, GR 711 10 Heraklion, Crete, Greece
{alfrag,vsiris,npetro}@ics.forth.gr

Abstract. In this paper we present and evaluate anomaly-based intrusion detection algorithms for detecting physical layer jamming attacks in wireless networks, by seeking changes in the statistical characteristics of the signal-to-noise ratio (SNR). Two types of algorithms are investigated: simple threshold algorithms and algorithms based on the cumulative sum change point detection procedure. The algorithms consider SNR-based metrics, which include the average SNR, minimum SNR, and max-minus-min SNR values in a short window. The algorithms are applied to measurements taken in two locations, one close and one far from the jammer, and evaluated in terms of the detection probability, false alarm rate, detection delay and their robustness to different detection threshold values. Our results show that the cumulative sum detection procedure can improve the detection probability and false alarm rate when measurements are taken far from the jammer, and can improve the robustness for different values of the detection threshold.

Keywords: Intrusion detection, signal-to-noise ratio, jamming, simple threshold algorithms, cumulative sum algorithms, performance evaluation.

1 Introduction

Due to their broadcast nature, wireless networks are more susceptible to attacks. Adversaries can exploit vulnerabilities in the physical and the medium access control layers [1,2,3,4,5], and heavily disrupt the communication between the network nodes. Attacks at the physical layer are usually referred to as jamming, and the corresponding adversaries (attackers) as jammers. A difficulty in detecting attacks at the physical layer is that channel impairments, which are not the result of attacks, can cause variations of the signal at a receiver.

* This work is supported in part by the European Commission in the 7th Framework Programme through project EU-MESH, ICT-215320, www.eu-mesh.eu

** Corresponding author.

*** V.A. Siris is also with the Department of Informatics at the Athens University of Economics and Business.

E. Osipov et al. (Eds.): WWIC 2010, LNCS 6074, pp. 192–203, 2010.

Intrusion detection involves the automated identification of unusual activity by collecting audit data, and comparing it with reference data. A primary assumption of intrusion detection is that a network's normal behavior is distinct from abnormal or intrusive behavior, which can be a result of a DoS (Denial-of-Service) attack. Various approaches to intrusion detection differ in the metrics (or measures/features) they consider, in addition to how and where these metrics are measured. Identifying the metrics to be monitored is important, because they affect the detection performance and the amount of monitored data can be particularly large, hence its collection can consume a significant amount of resources.

Intrusion detection procedures can be classified into three categories [6]: misuse (or signature-based) detection, anomaly detection, and protocol-based (or specification-based) detection. The three categories differ in the reference data that is used for detecting unusual activity: misuse detection considers signatures of unusual activity, anomaly detection considers a profile of normal behavior, and specification-based detection considers a set of constraints characterizing the normal behavior of a specific protocol or program.

In this paper we investigate anomaly-based intrusion detection algorithms. Unlike previous works, summarized in Sect. 2, that consider only very simple detection algorithms, a focus of this paper is to investigate and quantify when more intelligence in the detection algorithm can improve performance. In this direction, we consider both simple threshold detection algorithms and more advanced algorithms based on cumulative sum change point detection. The intrusion detection algorithms we consider seek changes in the signal-to-noise ratio (SNR) of received packets. In this work, we use a modified version of the madwifi driver[1] to perform SNR measurements. Our motivation for using SNR rather than some other metric (e.g., number of PHY errors) is that hardware radio interfaces and wireless device drivers typically provide values of the SNR for received packets.

The contributions of this paper are the following:

- We consider different metrics based on the SNR: average SNR, minimum SNR, and max-minus-min SNR.
- We investigate the performance of the algorithms in terms of the detection probability, false alarm rate, detection delay, and their robustness to different detection threshold values.
- We investigate the performance of the algorithms for measurements from an actual network, taken both close to and far from the jammer.
- We investigate the improvements of more intelligent algorithms, based on the cumulative sum (cusum) change point detection approach.

The rest of the paper is organized as follows. Related work is described in Sect. 2. The topology of the wireless testbed, the software used for collecting the SNR measurements, and the jamming model are presented in Sect. 3. The anomaly-based intrusion detection algorithms that we investigate are discussed in Sect. 4. The evaluation of the detection algorithms is presented in Sect. 5. Finally, conclusions and further work appear in Sect. 6.

[1] Madwifi project, `http://www.madwifi.com`

2 Related Work

This section presents a short overview of work related to intrusion detection in wireless networks. A key difference with this paper, which constitutes our main contribution, is that we investigate the performance improvements that can be achieved by cumulative sum change point detection algorithms, in terms of the detection probability, false alarm rate, detection delay, and robustness.

Different types of jammers have been defined in the literature. In [7], the authors describe four attack models: (*i*) a constant jammer that continuously emits RF signals without following any MAC protocol, (*ii*) a deceptive jammer that transmits normal packets, (*iii*) a random jammer that emits energy at random intervals followed by intervals of inactivity, and (*iv*) a reactive jammer that sends data only if it detects energy on the channel; this is the most effective jammer since it preserves energy, it can corrupt data with a higher probability, and is harder to detect. There are also jamming attacks at the MAC layer that try to exploit vulnerabilities in the RTC/CTS and NAV mechanisms [8]. In this paper, we consider a *periodic jammer* that periodically transmits packets, hence emits RF energy alternating between sleeping and jamming.

The authors in [9] describe several types of jammers and propose 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, while 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 detection delay are not presented. In [10], techniques that detect anomalies at all layers of a wireless sensor network are proposed. The authors show how the detection probability increases when the number of the nodes running the proposed procedure increases, but do not show the trade-off with the false alarm rate.

The authors of [11] show how the errors at the physical layer propagate up the network stack, and present a distributed anomaly detection system based on simple thresholds. A method for combining measurements using Pearson's Product Moment correlation coefficient is also presented. Several adversarial models are presented in [7], 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 introduce two detection algorithms, based on thresholds, that use signal strength and location information as a consistency check to avoid false alarms.

The authors of [8] presents 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 RTS/CTS and NAV mechanisms of the IEEE 802.11 protocol. The authors of [5] consider the sequential probability ratio test, applied however to detect MAC-layer misbehavior. Finally, cusum algorithms have been used in several detection systems [5,12,13,14] but they focus on the MAC and upper layers, while we consider a metric (SNR) in the physical layer.

3 Measurement System and Experimental Setup

3.1 Network Topology

The topology of the network used to collect the SNR measurements is shown in Fig. 1. All nodes are equipped with Mini ITX boards, with 512 MB RAM and a 80 GB hard disk. They are also equipped with Atheros NMP 8602 802.11 a/b/g wireless cards, controlled by the Madwifi v 0.9.4 MAC driver, on Ubuntu Linux.

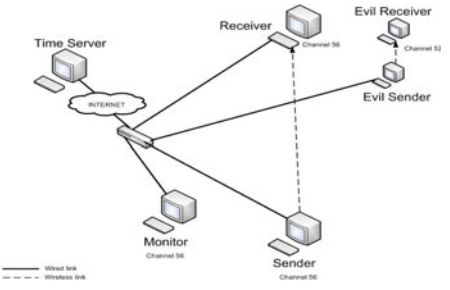


Fig. 1. Network layout

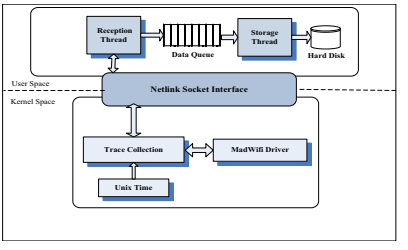


Fig. 2. Measurement module based on the Madwifi driver

UDP traffic is sent from the sender S to the receiver R at a constant rate of 18 Mbps. In addition to the receiver R, a monitor M also collects SNR measurements. Nodes S, M, and R all operate on channel 56. The network interface of monitor M is set to monitor mode, hence receives all packets sent on this channel. Jamming is performed by using two nodes, evil sender (ES) and evil receiver (ER); these two nodes operate on channel 52, which is adjacent to channel 56, hence produces interference. The solid lines in Fig. 1 represent a dedicated wired backbone for running the experiments. Finally, the time on all nodes is synchronized using the network time protocol (NTP) [15].

3.2 Collection of Measurements

Our modifications in the madwifi driver enabled the collection of SNR measurements in all the wireless nodes. The software collection module (Fig. 2) consists of several parts laying on both the kernel and user spaces of the Linux operating system. Data are collected through madwifi in kernel space and then asynchronously transmitted to user-space through the netlink socket interface. In user-space, the Reception Thread receives the data from kernel-space and stores them in a software 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 above threads, connected through the queue, is to increase performance because the copy of the data from memory to the hard disk does not block the reception of the new data coming from the kernel-space. The Storage and Reception threads execute independently in the multi-threaded environment of Linux.

With the same software we are capable of collecting additional information for each packet, such as the long and short retry counters, the retry field of the MAC header, and the timestamps of the outgoing or the incoming packets; the timestamps are necessary to time align measurements at the jammer (evil sender in Fig. 1) and at the nodes where SNR measurements are collected: the receiver and the monitor. If the transmitted or received frame is a data frame, the software also records information such as MAC addresses, IP addresses, port numbers, etc.

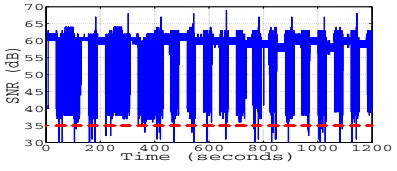


Fig. 3. Effect of jamming on SNR

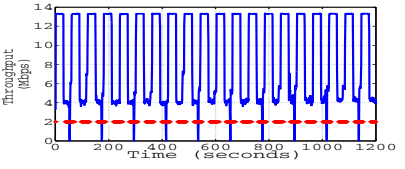


Fig. 4. Effect of jamming on throughput

3.3 Jamming Model

As indicated above, jamming is performed by an evil sender transmitting to an evil receiver on channel 52, which is adjacent with channel 56 on which the sender, receiver, and monitor operate, hence creates interference. The transmission power and rate of the evil sender is 13 dBm and 6 Mbps, respectively. The evil sender transmits data for 30 seconds, after which it remains inactive for 30 seconds. Figure 3 shows the impact of jamming on the SNR of the received packets. This figure shows that drops of the SNR value occur also in periods when there are no attacks. This is due to channel impairments but also to other sources of interference, since the network test-bed was located in a public area where people freely walked. Due to this behavior of the wireless channel, anomaly-based intrusion detection in wireless networks is challenging. Figure 4 shows the effect of jamming on UDP throughput.

4 Anomaly-Based Intrusion Detection Algorithms

We consider two types of algorithms: simple threshold algorithms and algorithms that implement a cumulative sum change point detection procedure. Both types can be applied to different metrics that are based on the SNR: average SNR, minimum SNR, and max-minus-min SNR. The values of these metrics are measured over a small time window.

4.1 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.

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

$$SNR_{min}(n) = \min_{n-K+1<i\leq n} SNR_i \,,$$

whereas the average SNR measured in window M is

$$\overline{SNR}(n) = \frac{\sum_{i=n-M+1}^{n} SNR_i}{M} \,.$$

An alarm is raised at the arrival of frame n if

$$\frac{\overline{SNR}(n)}{SNR_{min}(n)} \geq h \,, \tag{1}$$

where h is the detection threshold.

The Simple Max-Min algorithm. Rather than considering the minimum value of the SNR measured in a small window, this algorithm considers the maximum-minus-minimum values of the SNR measured in a small window. If D denotes the maximum-minimum value of the SNR, then

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

while 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

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

The Simple Average algorithm. This algorithm compares the average value of the SNR in a short window, with the average value in a long window. If $\overline{SNR_{short}}(n) = \frac{\sum_{i=n-K+1}^{n} SNR_i}{K}$ is the average value of the SNR in a short window K, then an alarm is raised if

$$\frac{\overline{SNR}(n) - \overline{SNR_{short}}(n)}{\overline{SNR}(n)} \geq h \,, \tag{3}$$

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

4.2 Cumulative Sum (Cusum) Algorithms

Cusum algorithms belong to the category of change point detection algorithms, and were first introduced in [16]. These types of algorithms have been widely used in the literature, e.g. [5,12,13,14]. Let $\{x\}$ be an independent and identically distributed random variable, and assume there are two hypotheses: $H_0 : x = x_0$ and $H_1 : x = x_1$. H_0 is true when there is no attack, while H_1 is true when there is an attack. The probabilities of the above hypotheses are p_{θ_0} and p_{θ_1}, respectively. The statistical distribution of x has probability p_{θ_0} before a change occurs (i.e., before a jamming attack), and p_{θ_1} after the change has occurred. Cusum aims to detect this change based on the log-likelihood ratio of $k - i$ subsequent observations from x_i to x_k, given by the formula [17]: $S_k^j = \sum_{i=j}^{k} s_i$, where $s_i = \log \frac{p_{\theta_1}(x_i)}{p_{\theta_0}(x_i)}$ is the log-likelihood ratio for the observations from x_i to x_k. Generally, there are two main categories of cusum algorithms: parametric and non-parametric. For the parametric cusum, a model for $\{x\}$ is required, which is not easy to obtain in the area of the communication networks and especially for the SNR due to the volatile nature of wireless networks. For this reason, we consider non-parametric cusum algorithms where a model of $\{x\}$ is not required.

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 of the algorithm, and $Z_n = \frac{\overline{SNR}(n)}{SNR_{min}(n)}$. Note that y_n in (4) increases as $Z_n = \frac{\overline{SNR}(n)}{SNRmin(n)} > a$, i.e. as the minimum SNR value is smaller than the average SNR value by some amount which is determined by the value of a. In the experimental evaluation, we have considered $a = 0.7$.

An alarm is signaled when

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

where h is the detection threshold.

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 SNR in the short time window, and the average maximum-minus-minimum SNR estimated in the long time window. In the experimental evaluation, we have considered $a = 8$.

The alarm rule for the cusum max-min algorithm is identical to the rule for the cusum min algorithm, and is given by (5).

The Cusum Avg (average) algorithm. As above, this algorithm has the same regression formula as the cusum min algorithm, given by (4), however Z_n is now given by

$$Z_n = \frac{\overline{SNR}(n) - \overline{SNR_{short}}(n)}{\overline{SNR}(n)}.$$

In the experimental evaluation, we have considered $a = 0.8$.

The alarm rule for the cusum max-min algorithm is identical to the rule for the cusum min algorithm, and is given by (5).

5 Performance Evaluation

This section presents the performance evaluation of the proposed algorithms using the SNR measurements collected at the receiver and the monitor, Fig. 1. Observe in this figure that the receiver is close to the jammer, whereas the monitor is far from the jammer. As expected, our results show that the algorithms perform better when measurements at the receiver are considered, compared to when measurements at the monitor are considered. The attacks are launched with the transmission of periodic traffic from the evil sender in Fig. 1, and are depicted as orthogonal boxes in Fig. 3.

The performance of the algorithms is evaluated in terms of the detection probability *DP*, false alarm rate *FAR*, average detection delay *DD*, and the robustness to different detection threshold values. The 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 the false alarms over the total duration of the experiment in minutes. A false alarm occurs when an alarm is raised but there is no attack. Finally, the average detection delay is given by $DD = \frac{\sum_{i=1}^{k}[t_a(i)-t_s(i)]}{k}$, where k is the total number of the attacks, $t_a(i)$ is the first time an alarm is raised during the i^{th} attack (more than one alarms can be raised during a single attack), and $t_s(i)$ is the time when the i^{th} attack begins. The detection delay is measured in minutes.

The size of the sliding window used in the experiments is 1000 and 10, for the long and short windows respectively. Note that 1000 samples correspond to approximately 0.9 seconds, which is about one thirtieth of the attack duration (30 seconds).

5.1 DP and FAR Tradeoff

We begin by discussing the tradeoff between the detection probability *DP* and the false alarm rate *FAR*, Fig. 5. The various points shown in this graph correspond to different values of the detection threshold parameter. Observe in the top three subfigures in Fig. 5, which shows the results when measurements at the receiver are used for detection, that the performance of all three algorithms, and their cusum variants discussed in the previous section, is very good and close to the top-left corner which corresponds to 100% detection probability and zero

false alarms/min for the false alarm rate. Hence, the tradeoff between detection probability versus false alarm rate is not influenced by the use of the cusum mechanism. This is not the case when detection is based on measurements at the monitor, which correspond to the bottom three figures in Fig. 5; observe that when detection is based on measurements taken at the monitor, which happens to be far from the jammer, the cusum mechanism can significantly improve the detection probability versus false alarm rate in the case of the *Max-Min* and *Min SNR* metrics. Finally, note that although for measurements at the receiver, the cusum mechanism does not improve the detection probability versus false alarm rate, as we will see in the following subsection, it does improve the robustness of the algorithms, where by robustness we mean that the performance of the algorithms remains relatively stable for small variations of the detection threshold.

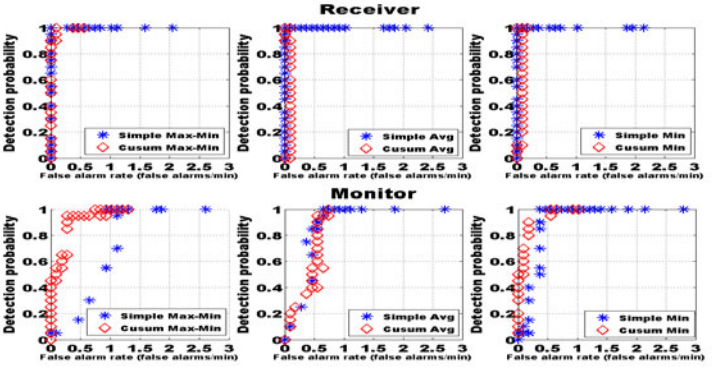


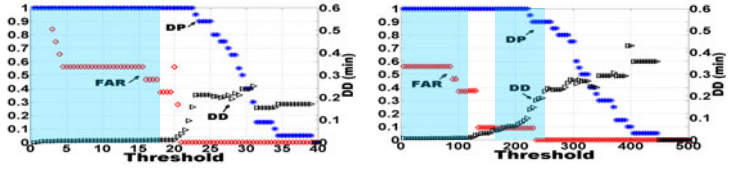
Fig. 5. Detection probability versus false alarm rate for the different algorithms when applied to measurements at the receiver and the monitor

5.2 Robustness and Detection Delay

Next, we discuss the robustness of the algorithms, by investigating how their performance, in terms of the detection probability and false alarm rate varies when the detection threshold changes. First, assume that M is defined as $M = \frac{1}{c+DP} + FAR$, $c > 0$, which combines the detection probability DP and false alarm rate FAR. We define that a detection algorithm is (relatively) robust if the detection threshold needs to change by more than 20% so that the metric M changes by more than 20%. Both these numbers and the specific form of function M are operator-dependent; the methodology we present to evaluate the robustness of the detection algorithms can be applied to any functions that combine the metrics DP, FAR. In the graphs shown below, the range of detection threshold values for which an algorithm is robust, is shown as a shaded area.

Max-Min algorithms. Figure 6 shows how the DP, FAR, and DD vary with the detection threshold. The values of DD are referred on the right vertical axis.

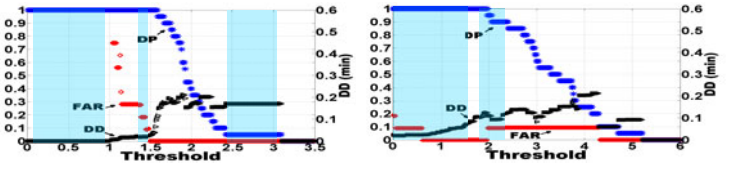
Also, in all subsequent figures, the values of *FAR* greater than one are not shown; these values appear for small values of the detection threshold located on the left part of each graph. Observe that the threshold values (21-22) where the *Simple Max-Min* algorithm achieves a high *DP* and low *FAR* are not inside the stable region. Hence, although for these values the *Simple Max-Min* algorithm achieves good performance in terms of the *DP* and *FAR*, its performance is sensitive to small changes of the detection threshold. On the other hand, the detection threshold (≈ 220) for which the *Cusum Max-Min* algorithm achieves high *DP* and low *FAR* is inside the stable region. Finally, observe that the detection delay for both the *Simple Max-Min* and *Cusum Max-Min* algorithms is less than 0.1 minutes, within threshold areas they achieve high performance.



(a) *Simple Max-Min* **(b)** *Cusum Max-Min*

Fig. 6. *DP*, *FAR*, and *DD* for *Simple Max-Min* and *Cusum Max-Min*; measurements at the receiver

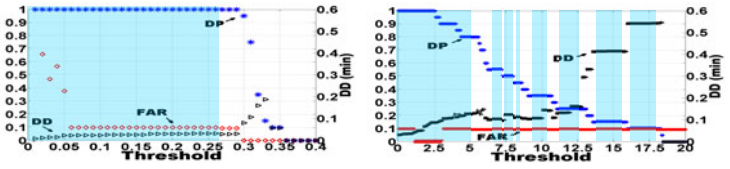
Min algorithms. Similar to the *Max-Min* algorithms, the *Cusum Min* algorithm achieves a high *DP* and low *FAR* for values of the detection threshold that are inside the stable region, which is not the case for the *Simple Min* algorithm, Fig. 7. The detection delay for the *Simple Min* algorithm is approximately 0.002 minutes, whereas for the *Cusum Min* the detection delay increases from 0.003 to 0.009 minutes, as the detection threshold increases.



(a) *Simple Min* **(b)** *Cusum Min*

Fig. 7. *DP*, *FAR*, and *DD* for *Simple Min* and *Cusum Min*; measurements at the receiver

Avg algorithms. Fig. 8 shows the *DP*, *FAR*, and *DD* for the *Simple Avg* and *Cusum Avg* algorithms, as a function of the detection threshold. As with the *Max-Min* and *Min* algorithms, the cusum procedure improves the robustness of the algorithm. The detection delay for the *Simple Avg* algorithm is approximately 0.05 minutes, whereas for the *Cusum Avg* algorithm the delay varies from 0.05 to 0.1 minutes.



(a) *Simple Avg* **(b)** *Cusum Avg*

Fig. 8. *DP*, *FAR*, and *DD* for *Simple Avg* and *Cusum Avg*; measurements at the receiver

All the above algorithms except the *Cusum Max-Min* can achieve perfect performance: 100% detection probability and zero false alarm rate.

6 Conclusions and Further Work

We have presented and evaluated algorithms for the detection of jamming attacks in IEEE 802.11 networks, based on actual measurements. The algorithms seek changes in the statistical characteristics of the collected SNR values and are of two types: simple threshold algorithms and cusum-type algorithms. The algorithms were evaluated in terms of the detection probability, false alarm rate, average detection delay and their robustness to different detection threshold values.

All algorithms have high performance when applied to measurements collected at a node close to the jammer. In the case of measurements collected at a node far from the jammer, the performance of the simple threshold algorithms deteriorates significantly; for such measurements the algorithms based on cumulative sum change point detection show higher performance in terms of the detection probability and false alarm rate. Even in the case of measurements taken close to the jammer, the cusum algorithms have higher robustness, i.e. they exhibit similar performance for a range of value of the detection threshold.

Ongoing work includes the investigation of approaches for combining measurements at different locations, and at different layers.

References

1. Bicacki, K., Tavli, B.: Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks. In: Computer Standards and Interfaces. Elsevier, Amsterdam (September 2008)
2. Thuente, D., Newlin, B., Acharya, M.: Jamming Vulnerabilities of IEEE 802.11e. In: Proc. of MilCom 2007, pp. 1–7. IEEE, Los Alamitos (October 2007)
3. Wenyuan, X., Ke, M., Trappe, W., Yanyong, Z.: Jamming sensor networks: attack and defense strategies. IEEE Network, 41–47 (May 2006)
4. Hall, M., Silvennoinen, A., Haggman, S.: Effect of pulse jamming on IEEE 802.11 wireless LAN performance. In: Proc. of MilCom 2005, pp. 2301–2306. IEEE, Los Alamitos (October 2005)

5. Cardenas, A., Radosavac, S., Baras, J.: Evaluation of Detection Algorithms for MAC Layer Misbehavior: Theory and Experiments. To appear in IEEE/ACM Transactions on Networking (2009)
6. Mishra, A., Nadkarni, K., Patcha, A.: Intrusion Detection in Wireless Ad Hoc Networks. IEEE Wireless Communications, 48–60 (February 2004)
7. Xu, W., Trappe, W., Zhang, Y., Wood, T.: The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In: Proc. of ACM MobiHoc (May 2005)
8. Thamilarasu, M., Mishra, S., Sridhar, R.: A Cross-layer approach to detect jamming attacks in wireless ad hoc networks. In: Proc. of Milcom 2006, Washington DC, USA, October 2006, pp. 1–7 (2006)
9. Cakiroglou, M., Ozcerit, T.: Jamming detection mechanisms for wireless sensor networks. In: Proc. of 3rd Int. Conference on Scalable Information Systems, Napoli, Italy (June 2008)
10. Bhuse, V., Gupta, A.: Anomaly intrusion detection in wireless sensor networks. Journal of High Speed Networks, 33–51 (2006)
11. 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 (2006)
12. Wang, H., Zhang, D., Shin, K.: Change-point monitoring for the detection of DoS attacks. In: Transactions on Dependable and Secure Computing. IEEE, Los Alamitos (2004)
13. Yan, G., Xiao, Z., Eidenbenz, S.: Catching instant messaging worms with change-point detection techniques. In: Proc. of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats (April 2008)
14. Chen, Y., Hwang, K., Ku, W.: Distributed Change-Point Detection of DDoS Attacks: Experimental Results on DETER Testbed. In: Proc. of USENIX Security Symposium, Boston, USA (August 2007)
15. Ntp: the network time protocol, `http://www.ntp.org`
16. Page, E.S.: Continuous inspection schemes. Biometrika, 100–115 (1954)
17. Basseville, M., Nikiforov, I.V.: Detection of Abrupt Changes: Theory and Applications. Prentice-Hall, Englewood Cliffs (1993)