Machine Learning-based Jamming Detection for IEEE 802.11: Design and Experimental Evaluation

Oscar Puñal*, Ismet Aktaş*, Caj-Julian Schnelke,
Gloria Abidin, Klaus Wehrle
Communication and Distributed Systems
RWTH Aachen University, Germany
Email: {firstname.lastname}@rwth-aachen.de
*Co-primary author

James Gross
School of Electrical Engineering
KTH Royal Institute of Technology, Sweden
Email: james.gross@ee.kth.se

***Abstract*—Jamming is a well-known reliability threat for mass-market wireless networks. With the rise of safety-critical applications this is likely to become a constraining issue in the future. Thus, the design of accurate jamming detection algorithms becomes important to react to ongoing jamming attacks. With respect to experimental work, jamming detection has been mainly studied for sensor networks. However, many safety-critical applications are also likely to run over 802.11-based networks where the proposed approaches do not carry over. In this paper we present a jamming detection approach for 802.11 networks. It uses metrics that are accessible through standard device drivers and performs detection via machine learning. While it allows for stand-alone operation, it also enables cooperative detection. We experimentally show that our approach achieves remarkably high detection rates in indoor and mobile outdoor scenarios even under challenging link conditions.**

I. Introduction

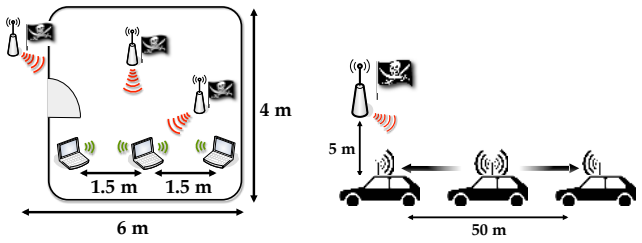
Jamming attacks consist of radio signals maliciously emitted to disrupt legitimate communications. Various studies show this in the context of 802.11 and 802.15.4 systems [4], [17], [24], as well as in the context of cellular networks [9], [18]. With the proliferation of (time-critical) machine-to-machine applications in general, and safety-critical applications in vehicular ad-hoc networks (VANETs) in particular, the importance of jamming-aware communications is expected to increase in the future. In general, the impact of jamming can be alleviated by either increasing the robustness of the legitimate signal [11], [15] or by migrating the communication to a different frequency band [14]. However, many of the proposed countermeasures cannot always be applied on already existing systems and, in most cases, the only alternative is to try to detect the jammer. In the context of safety-critical communications over VANETs, the detection of a jamming attack could, for instance, alert the driver about potentially malfunctioning applications.

Jamming detection can be performed by dedicated devices or by algorithms within the communication devices themselves. In general, the latter case is associated with less overhead and costs. In either case, one has to rely on previously acquired knowledge of the communication behavior under *normal* and *jammed* conditions. This requires the tracking of potential indicators (or metrics) of jamming activity, which are obtained at different layers (e.g., packet delivery rate at the application layer and channel busy time at the MAC layer). The use of a cross-layer architecture can ease the task of collecting necessary metrics and, hence, of jamming detection [15].

In literature, only few experimentally-evaluated approaches for jamming detection have been proposed [7], [24], [23], which either do not explicitly address 802.11 communications [7], [24] or focus on very specific, and hence hardly generalizable, jamming attacks [23]. Common approaches *manually* set thresholds for the selected metrics based on empirical observations [7], [24]. However, during normal operation other effects such as network congestion and challenging wireless link conditions can exhibit a similar impact as jamming, which degrades the detection accuracy. Furthermore, adding more metrics, which theoretically increases the accuracy, complicates the problem of the manual threshold setting.

In this paper, we present a machine learning-based jamming detection approach for 802.11 networks that weighs and combines a considerable set of metrics and automatically selects appropriate thresholds, thereby circumventing the arduous and error-prone manual tuning. Our approach relies on metrics available from drivers of commodity network interface cards. For convenience, we utilize CRAWLER [3], a cross-layer tool that facilitates the access to the metrics. Afterwards, the metrics are provided to a machine learning algorithm to predict the likelihood of a jamming attack. The proposed approach features a high detection accuracy in different scenarios (indoor and vehicular), under different propagation conditions (good- and bad-link conditions, with and without concurrent traffic from neighbor networks), and for two different jammer types (constant and reactive). In addition, our approach easily integrates cooperative jamming detection to further improve the accuracy without incurring significant costs.

The remainder of this paper is organized as follows. In Section II, we introduce metrics for jamming detection and analyze their reaction to jamming. Section III presents the design of our machine learning-based jamming detection approach, which is evaluated in Section IV. In Section V we discuss practical problems. An overview of related work is provided in Section VI. Finally, in Section VII we conclude our work and discuss on future work directions.



(a) Indoor Scenario (b) Outdoor Scenario

Fig. 1. Indoor (reference) and outdoor scenarios considered in the evaluation.

II. Challenges of designing jamming detection strategies

To differentiate jamming from normal operation, it is necessary to analyze the impact of jamming on the system performance. Therefore, we investigate a set of *metrics* that react to jamming attacks and *helper metrics* that do not show a reaction to jamming, but provide context for an appropriate weighting of other metrics. Our results demonstrate the difficulty of jamming detection by showing the complex interdependencies between the scenario, the system behavior, and the jammer.

A. Reference Scenario and Measurement Setup

The scenario used to evaluate the suitability of the selected metrics was an office room located in the UMIC Research Centre at the RWTH-Aachen University, which is sketched in Figure 1(a). Our setup consisted of three Linux PCs equipped with 802.11g Atheros WLAN cards running the ath9k driver [1]. The three nodes were configured in ad-hoc mode and communicate on channel 11 in the 2.4 GHz band, which was not occupied by any other network during our experiments.

In order to mimic ideal and challenging link characteristics, we considered two different configurations which we refer to as **good-link** and **bad-link**. In the good-link configuration, the nodes were placed close to each other and the transmission was parameterized to achieve, on average, a high packet delivery rate. The bad-link topology was characterized by a poor communication performance, which was achieved by selecting a lower transmit power and/or by adding attenuation elements at the output of the radio front-end. For each configuration we collected data under **normal** and **jammed** conditions. In the latter case, we placed the jammer at different positions (cf. Fig. 1(a)) and varied its output power to impact the performance of the communicating nodes differently. We implemented the jammer on a WARP board [13], which provides an 802.11-like OFDM physical layer featuring a 10 MHz bandwidth and an output power of 18 dBm in the 2.4 GHz band. The jamming signal consisted of a preamble and BPSK modulated random payload of variable length. The jamming signals prevented the legitimate devices to access the medium, which differs from what has been reported for other Atheros cards in [21], [17].

Constant jammer: Implementing a constant jammer on WARP is not entirely possible, since the amount of time that the boards can be transmitting a single signal is upper-bounded. We measured it using a spectrum analyzer to be about 2.7 ms. Between two consecutive signals there is a 10 μs gap required by the hardware to set up a new transmission. Nevertheless, this marginal *off-phase* is expected to not affect the performance of the legitimate communication, as this gap is not large enough for 802.11 stations to access the medium.

Reactive jammer: The reactive jammer starts a transmission when it senses energy on the channel above a threshold regardless of the type of signal detected. We set the threshold to -65 dBm to achieve a sufficiently high jammer sensitivity, while guaranteeing a low number of false detections, that is, avoid reacting to signals from neighbor 802.11 networks or other sources of electromagnetic activity. The jammer has a total reaction delay of 12 μs. This is fast enough to partially interfere the preamble of the 802.11 signal, which is known to increase the effectiveness of the attack [8].

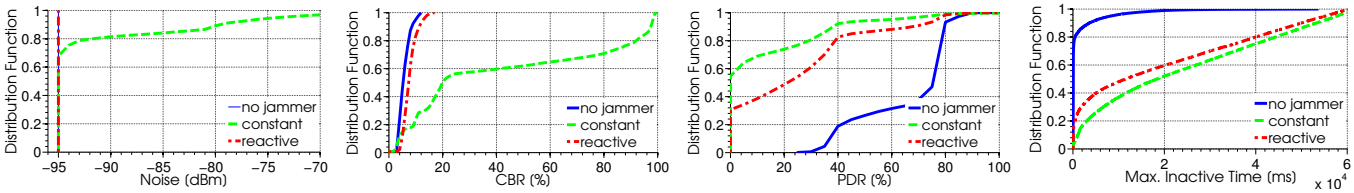
B. Experimenting with Indicators of Jamming Activity

We experimented with multiple metrics to detect jamming activity. Candidate metrics were selected based on two main criteria. First, we focused on metrics that are accessible via a common driver of commodity 802.11 network interface cards. Second, the metrics should work regardless of the type of traffic exchanged by the nodes. For instance, we discarded the number of *frame retransmissions*, since this metric requires the use of ACK frames that are not available in broadcast transmissions. Finally, we chose six metrics for further analysis, which we divided into three categories: (i) channel, (ii) performance, and (iii) signal metrics.

Channel metrics: These metrics sample the state of the wireless channel. We identified *noise* and *channel busy ratio (CBR)* as relevant. Noise is defined as the power measured on the channel during idle times of the transceiver [2]. Jamming signals that are transmitted while the legitimate nodes are idle (e.g., constant jammer) are likely to be included in the noise measurements of the cards as shown in Figure 2(a). However, a minimum jamming power and interference duty cycle are required for the cards to include the jamming signal into the noise measurements [19]. This happened to only 30% of the constant jammer samples collected in our indoor experiments (see Figure 2(a)).

The CBR measures the time (normalized to the observation time) that the wireless channel has been sensed busy. The channel is considered busy if the received power is above the clear channel assessment (CCA) threshold. As reactive jamming attacks are launched once the legitimate nodes have gained access to the medium, no impact is expected from this jammer on noise and CBR metrics, which can be clearly observed in Figures 2(a) and 2(b).

Performance metrics: This type of metrics can only be obtained if a connection is established between two or more stations. We identify *inactive time (IT)* and *packet delivery ratio (PDR)* as suitable metrics. The IT corresponds to the time that elapses between two consecutive successful packet receptions, including probing, beacons, and payload frames. Specifically, we account for the maximum IT at a node measured over the links to its neighbors.



(a) Noise (b) CBR (c) PDR (d) MAX IT

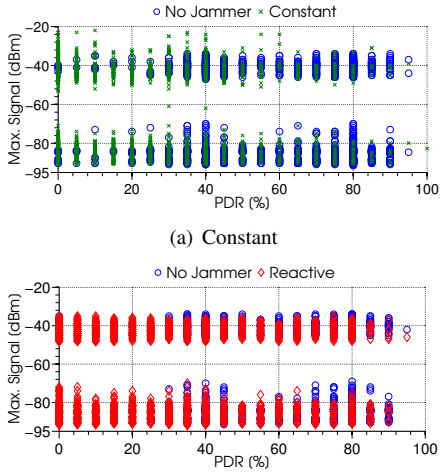
Fig. 2. CDF to compare the impact on all selected metrics of the **constant** and **reactive jammer** with the **non-jammed** case. Noise is completely unaffected by the reactive jammer, in contrast to the constant jammer. The CBR is strongly affected by the constant jammer, while the reactive jammer has only a marginal impact on it. PDR and max. IT are significantly affected by the presence of both jammers.

As opposed to the other metrics, the PDR is not directly provided by the card. For its computation, each node is aware of the number of network members in its hearing range and of a predefined rate for generating probing packets. With that knowledge, and based on the number of correctly received probing packets, the PDR can be computed. Figures 2(c) and 2(d) show that these metrics are good indicators for detecting jamming activity, since, in most cases, they clearly separate jamming from normal operational conditions.

Signal metrics: Signal is the power measured upon arrival of a packet, but only passed to higher layers in case of successful reception. This metric is a *helper* metric, that provides a useful context (i.e., link quality) to the PDR and the max. IT metrics, although it is not explicitly affected by jamming. For instance, a low received signal power is likely to result in a low PDR even if the jammer is silent. This knowledge is important to appropriately weigh the significance of PDR and max. IT accordingly. In our experiments, instead of collecting a single signal metric (i.e., the average power), we have observed that the differentiation between *minimum* and *maximum* signal over all links is most valuable.

C. Threshold Identification

A common strategy in related work is to manually choose thresholds for selected metrics [24] based on their behavior in a specific scenario. However, it is a difficult task to appropriately separate the values of the metrics and weigh them based on their significance. We illustrate this issue by jointly collecting samples of measured PDR and received signal strength (plus measured noise power). We collect these samples in our reference scenario without jamming activity, following the approach proposed in [24]. These samples correspond to the blue circles depicted in Figure 3. We then manually determine the thresholds to best capture normal operation, specifically the thresholds are set so as to contain 99% of the unjammed samples as in [24]. Next, we activate the jammer and evaluate how well this method can identify jamming activity. In Figures 3(a) and 3(b) we observe a clear overlap of jammed and unjammed samples, which anticipates inaccurate detection rates. From these figures we derive two major observations: (1) metrics for jamming detection proposed by related work in the context of general wireless networks do not necessarily work well in 802.11 and (2) finding appropriate thresholds, even for only two metrics, is already a difficult task. The combination of multiple metrics will drastically increase the complexity



(a) Constant

(b) Reactive

Fig. 3. Consistency check approach of PDR vs. max. signal in indoor scenario as proposed by Xu et al. in [24]. Samples for jammed and not jammed overlap, which makes a clear threshold identification impractical for 802.11.

and make manual threshold setting impractical. For tackling this complex problem, we use machine learning algorithms, as they are known to be well-suited for multi-dimensional (binary) classification problems such as the decision about the presence and absence of jamming based on multiple metrics.

III. Detection System Design

Our jamming detection approach consists of two phases: (1) the collection of training data and (2) the application of machine learning on the collected data.

A. Data Collection Phase

Our machine learning algorithm takes training data as input. Therefore, the selected metrics need to be accessed and forwarded to the machine learning component, as illustrated in Figure 4. While most of the metrics are provided by the 802.11 device driver, the PDR is obtained from the application layer. As the latter requires a supervised packet exchange mechanism, we have incorporated an information exchange component into our design. The other metrics reside in the kernel space of the operating system, but they have to be provided to the user space in order to be used by the machine learning component. To address this task, we incorporate a cross-layer component (see ① in Fig. 4) that improves the flexibility of the framework and reduces the complexity. The

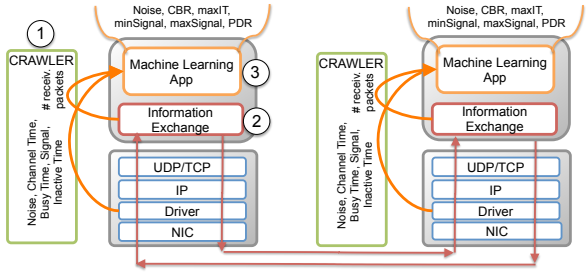


Fig. 4. Design overview of our detection approach consisting of three components: (1) Metrics are accessed via CRAWLER and are provided to the machine learning component. (2) Similarly does the information exchange component with the PDR metric. (3) Based on the gathered data, the machine learning component decides on the presence of the jammer.

details of the cross-layer and information exchange components are described in the following.

Cross-layer component: A cross-layer component should (i) offer the flexibility to include and prepare diverse metrics from different protocol layers and (ii) still reduce the complexity, i.e., simplify the access to protocol and system information without requiring excessive effort and knowledge about system details. As the cross-layer architecture CRAWLER [3] offers these features, we incorporated it into our framework. CRAWLER is an open-source software for Linux that allows cross-layer developers to express their monitoring and optimization requirements in an abstract and declarative way. CRAWLER provides many accessors to read and write system information ranging from TCP-IP to our metrics partially gathered directly from the WLAN devices.

Information exchange component: We propose the exchange of probing packets between nodes to measure the PDR (see ② in Fig. 4). We have implemented the packet exchange in a client-server manner running in the user space of the operating system. Each node runs the server and the client. The client broadcasts UDP packets every 100 ms. These packets have a total size of 57 Byte. In particular, 8 bit are reserved for the message type, although one message type is currently used, we reserved these bits for future use. A 16 bit value can be utilized to enable a cooperative mode to convey the detection probabilities from neighbor nodes. We later show in our evaluation that the use of the cooperative mode increases the jamming detection accuracy significantly (cf. Section IV-F). Finally, 54 Byte are necessary for protocol headers and CRC checksums. Hence, the broadcast of probing packets introduces a per-station overhead of about 570 Byte/s.

B. Machine Learning Phase

Before using machine learning for detecting the presence of a jammer, training data needs to be collected and provided to our machine learning component (cf. ③ in Fig. 4) for learning. Our training data consists of multiple instances of the decision problem, which are themselves divided into input variables or *features* (i.e., the six selected metrics) and a corresponding output variable or *class* (i.e., a binary variable stating whether the jammer is active or not).

Learning: In this work we have considered multiple learning algorithms, which are introduced and evaluated in Section IV-G. However, most investigations exclusively employ Random Forests [6], a sophisticated decision tree-based classifier known to be superior, in terms of accuracy, to most other classifiers [6]. For learning, Random Forests generates a large number of random decision trees (we empirically determined 50 trees with a depth of 10 to be a good trade-off and we later use this dimensionality in the evaluation part), the so-called *forest*. The input variable and the splitting threshold chosen at a node are automatically selected so as to maximize the classification accuracy. Finally, the leaf nodes represent the distribution of values that the output variable takes for the corresponding path through the decision tree.

Predicting: During operation, the input variables are continuously monitored and new instances (i.e., new values of metrics) are pushed down each decision tree reaching a specific leaf node. Depending on the distribution of the output variable at the leaf node, the tree will either *vote* for the presence of a jammer (i.e., output a *one*) or against it (i.e., output a *zero*). Finally, the votes of all trees are aggregated into a single output variable representing the **prediction probability** of jamming. In its default configuration, jamming activity is assumed if the predicted probability is larger than 0.5.

IV. Evaluation

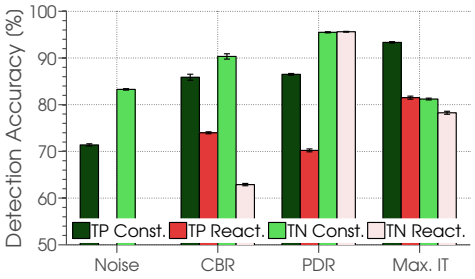
In earlier sections we have pointed out the necessity of appropriately selecting, combining, and weighting metrics to identify jamming attacks. In this section we present a detailed evaluation of our proposed approach in a representative set of scenarios to underline our arguments.

A. Measurement Methodology

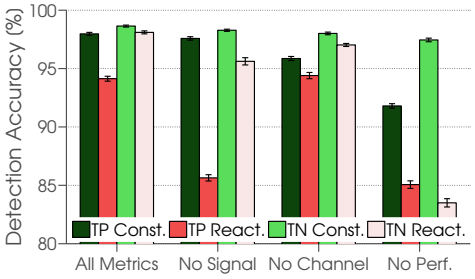
We have conducted static indoor and mobile outdoor experiments. We first provide results for the indoor tests. The experimental settings for the evaluation are the same as in the reference scenario introduced in Section II-A. We have configured the nodes to gather the value of the metrics every second. For every chosen topology, we conduct multiple runs with a duration of 60 s. The value of the binary output variable (i.e., the jammer activity) is introduced off-line once the measurement is finished. We collect the same number of instances with and without jammer activity in order to avoid biased learning. In the reference scenario, we collect a total of 27000 samples, namely 9000 for each jammer and 9000 without jammer. From the final training set, we randomly select 60% for learning and 40% for testing. To minimize the impact of this selection, we run the learning algorithm 20 times considering a different subset of samples at each iteration. The prediction accuracy is obtained on the samples reserved for testing. Unless specified differently, our results show the average detection accuracy together with the 95% confidence intervals.

B. Detection Accuracy

In our evaluation we show the **true positive (TP)** rate, i.e., the correct detection of existing jammer activity, and the **true negative (TN)** rate, i.e., the correct identification that



(a) Prediction accuracy achieved with a single metric. Although each metric reacts to jamming, relying on a single metric is not sufficient for reliable detection.



(b) Prediction accuracy obtained with all metrics and when a certain group of metrics is excluded from learning. As expected, the combination of all metrics achieves the highest detection accuracy.

Fig. 5. Detection accuracy obtained in the indoor reference scenario.

there is no jammer. In Section II-B we have shown that four metrics provided by commodity cards are suitable indicators of jamming activity. However, it is unclear if a subset of these metrics (or even a single metric) is able to yield the desired accuracy. Figure 5(a) shows the detection accuracy obtained in the reference scenario if only one metric is considered for learning. It can be observed that every metric has the potential of detecting jamming activity, which is illustrated by TP rates larger than 50% with the single exception of the noise metric in case of reactive jamming. The latter is evident based on the observations of Section II-B. Hence, relying on a single metric is not sufficient for guaranteeing a reliable detection.

Figure 5(b) shows the accuracy achieved with all available metrics compared to the accuracy when certain metrics are excluded. As expected, employing all metrics results in higher detection rates. The *no channel* group (i.e., excluding noise and CBR) yields a high accuracy, although the detection of the constant jammer is slightly worse. Excluding the *signal* metrics degrades the detection of reactive jamming. Clearly, the performance metrics (i.e., PDR and max. IT) are most important, as excluding them from the learning phase degrades all detection rates significantly. This was expected based on the results presented in Section II-B. To summarize, although single metrics can be used to some extent to detect a jammer, a holistic consideration of multiple metrics is the right strategy towards an efficient jamming detection.

C. Impact of Concurrent 802.11 Traffic

Besides jamming there are other sources of interference that can impact 802.11 communication, thereby complicating the detection of an attack. In this context, we are interested in evaluating the ability of our approach to detect jamming activity in the presence of intense traffic generated by a neighbor 802.11 network. For that, we placed two additional nodes in the reference scenario that communicated with each other in an ad-hoc fashion. Each node run the *iperf* application to generate an average traffic load of 12 Mbit/s with a fixed MTU size of 1500 Byte. The nodes were located close (about 2-3 m) to the original three-nodes and used the same frequency channel for transmission.

TABLE I
Detection Accuracy with & without Concurrent WLAN Traffic

	Constant		Reactive	
	TP	TN	TP	TN
(1) Without concurrent Traffic	97.97	98.64	94.13	98.10
(2) Concurrent Traffic 12 Mbit/s	98.44	99.70	94.31	99.00
(3) Training 1 for predicting 2	98.05	72.70	89.36	54.34
(4) Training 1&2 for predicting 2	98.23	99.72	93.18	99.19

Row 2 in Table I shows the detection accuracy achieved by our approach when concurrent 802.11 traffic is present during the learning phase and later also during prediction. For better readability we omit the 95% confidence intervals, which are always below 1%. It can be observed that the detection rates for both jammers are not degraded compared to the ones obtained without background traffic (as in Row 1), which indicates that our approach efficiently differentiates between jamming and 802.11 interference. However, the activity of legitimate interference can have an unpredictable pattern depending on the number of neighbor nodes and the amount of traffic they generate. Performing learning without accounting for concurrent traffic leads to a significant drop in accuracy if this traffic activity appears only during the detection (see Row 3 in Table I). To overcome this problem, it is important to collect training data samples under different conditions that are likely to emerge during operation. We show (in Row 4 of Table I) that by combining training data samples from different scenarios (i.e., from Rows 1 and 2), high TN and TP rates are obtained, which are comparable with the accuracy achieved with scenario-specific learning.

D. Impact of Outdoor Mobility

Safety-critical communications in vehicular scenarios have tough reliability constraints and strict delivery deadlines. Hence, a responsive and accurate jamming detection is important to initiate appropriate countermeasures. We evaluated our approach in an outdoor scenario with mobility, which is illustrated in Figure 1(b). We placed two cars at the ends of a parking lot, while mobility was introduced by a third car that was moving back and forth between the static nodes at a maximal speed of 25 km/h. The jammer was located close to one of the static nodes. The wireless link between the static nodes was characterized (without jammer activity) by a low PDR of about 40% that dropped further due to the attenuation caused by the moving vehicle. Depending on the position of the latter, the quality of the links varied

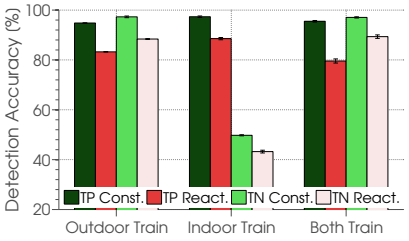


Fig. 6. Outdoor scenario: Detection accuracy when using outdoor training data (Outdoor Train) compared to training with indoor data for outdoor prediction (Indoor Train) and combining both training data sets for outdoor prediction (Both Train). Collecting training samples under different conditions is a requisite to achieve high detection accuracy.

significantly and we obtained PDR values that spanned the whole range. We conducted multiple runs of 60 s and collected a total of 4500 data samples (1500 for each jammer and 1500 without jammer). Figure 6 (cf. Outdoor Train) shows that the constant jammer can be detected with high reliability, while the accuracy for the reactive jammer is slightly above 80%. In general, node mobility and the signal propagation characteristics of the outdoor environment do not hinder an accurate detection of the jammers.

Obtaining training data in outdoor environments can be time-consuming. We have investigated the reusability of indoor learning for detecting outdoors. Figure 6 (cf. Indoor Train) shows that the TN rates for both jammers are degraded and fall below 50%. In general, differences in the scenario characteristics, and hence in the behavior of the metrics, lead to notable training dependencies. Therefore, as already discussed in Section IV-C, collecting training samples under different conditions is a requisite for robust and flexible jamming detection. This is underlined in the figure by the high accuracy achieved when the learning is applied on training data that contains both indoor and outdoor samples (cf. Both Train).

E. Exploiting Detection History

So far, every node decides (with one second granularity) on the presence of a jammer using instantaneous information. However, we have observed that the predicted jammer probability features a certain degree of correlation in the time domain. Figure 7(a) is a 10 s excerpt of a measurement in the outdoor scenario when the constant jammer was active. The figure illustrates the time correlation of the probability as predicted by Random Forests and the final decision about the presence of a jammer. At some points, the probability falls below the 0.5 threshold and a wrong decision is made.

Figure 7(b) shows the burst length of erroneous decisions in the outdoor scenario. It can be observed that the majority of the detection errors are *isolated events* (i.e., they are preceded and followed by correct detections) and that more than five consecutive detection errors rarely happen. We have identified small fluctuations of the signal strength under bad-link conditions, as the main cause for isolated errors. These observations can be exploited for more efficient detection approaches. We have investigated the benefits of combining successive predictions to intercept these single detection errors. Specifically, we apply a moving average (*mAvg*) of a particular window size to account for past probabilities. For a window of size k, the values of $k-1$ previous detection probabilities are stored and combined with the current probability. Figure 7(c) shows the benefits provided by this method as a function of the window size. Remarkable is the 10% higher detection accuracy achieved with a window of 3 s, as this size is able to efficiently intercept the isolated detection errors. Further increasing the window size provides only a moderate gain and can even degrade the accuracy.

F. On-the-Fly Jamming Detection:

The evaluation results provided so far have been obtained off-line by applying our detection algorithm on previously collected data samples (not used for learning though). However, having an approach that enables immediate predictions is mandatory for real-world applicability. We have implemented our learning-based jamming detection framework on the 802.11 devices to perform *on-the-fly* predictions. This is achieved by installing the required machine learning libraries (e.g., we used OpenCV 2.4.3 for Random Forests) on the devices and redirecting the collected metrics to the machine learning component (cf. ③ in Fig. 4). For the on-the-fly detection to work, the outcome of the learning phase (e.g., the structure of the forest) needs to be stored and made accessible to the machine learning component. In the following, we evaluate our framework in on-the-fly mode.

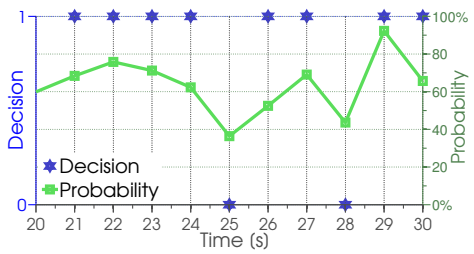
Scenario Details and Methodology: We consider an office room ($25\,m^2$) located in the building of the Communication and Distributed Systems Chair. The topologies chosen in this scenario were comparable to those of the reference scenario (cf. Figure 1(a)). Nevertheless, small differences in the propagation conditions of both scenarios were observed, especially with respect to the background 802.11 activity.

The jammer was placed at different locations within the room so as to affect the communication differently. We measured for a total of 60 minutes (with and without jammer) on different days and at different working hours. It is important to note that no specific learning was conducted, instead, the outcome of the learning obtained in the reference scenario was reused for detection in this new one. In the following we show the results for our on-the-fly detection (referred to as basic approach) and for two enhancements that increase the detection accuracy.

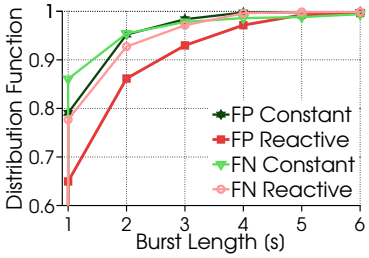
Results basic approach: Figure 8 shows the evaluation results for both jammers. In general, a high detection accuracy is achieved, even without the availability of scenario-specific learning. For instance, the TN rates are above 85% for both jammers and the TP rate for the constant jammer is close to 100%. The detection of the reactive jammer is in general lower, but the TP rate is still above 85%.

Advanced approaches: We extend the basic approach to incorporate the moving average mechanism (i.e., *mAvg*) to exploit the correlation in the time domain. Figure 8 shows the benefits of using moving average (window size of 3 s).

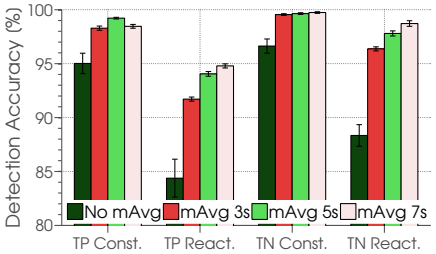
We are also interested in exploiting the correlation in the space domain, as nodes that are close to each other can



(a) Time evolution of the detection decision and probability (excerpt from outdoor measurement). A high correlation between consecutive probabilities and the presence of isolated errors can be observed.

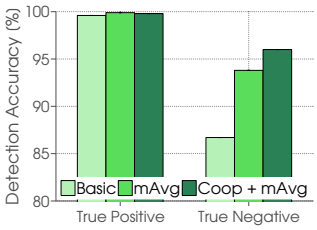


(b) Number of consecutive false predictions in outdoor scenario helps to determine the size of the moving average.

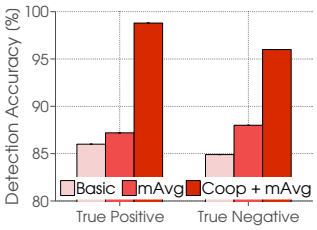


(c) Exploiting detection history (mAvg) improves detection in the outdoor scenario.

Fig. 7. Based on observed temporal correlations in the detection probability (cf. Fig 7(a)) and large proportion of *isolated* errors (cf. Fig. 7(b)), we propose a moving average to improve the detection accuracy (cf. Fig. 7(c)).



(a) Constant Jammer.



(b) Reactive Jammer.

Fig. 8. Benefits of moving average (mAvg) and cooperation (Coop) for an accurate jamming detection. Despite considering different scenarios for learning and testing, the combined approach (Coop + mAvg) achieves remarkably high detection rates.

be expected to be similarly affected by the jammer. This information redundancy can be exploited by letting neighbor nodes exchange their individually computed detection probabilities and can be easily conveyed (after 1 s delay) within the probing packets without significantly adding complexity or overhead. We enabled our information exchange component to support *cooperative jamming detection* (i.e., *Coop*). However, we have investigated only a *naive* approach, where the nodes assume that all network members within hearing range are identically affected by the jammer and correspondingly they *average* the detection probabilities of all neighbors. Smarter approaches that, for instance, make use of GPS information are more appropriate to weigh these probabilities. Note that this investigation is out of the scope of this paper.

Results advanced approach: In general, the use of moving average alone already improves the detection accuracy. Figure 8 shows that the TN rate for the constant jammer increased by up to 7%. The figure also shows the improvements of cooperation combined with moving average (i.e., *Coop + mAvg*). This combined approach shows the most significant gain with up to 11% higher accuracy compared to the basic approach, which brings all detection rates above 95%.

Conclusion: We demonstrated the ability of our jamming detection framework to achieve a high detection accuracy at runtime. In addition, we observed that the basic approach achieves a high detection accuracy even when using learning based on training data obtained in a different scenario. This fact highlights the reusability of the learning phases to be applied on scenarios of similar characteristics. Furthermore, the proposed advanced mechanisms that exploit correlation in the time and space domains, in particular when applied together, achieve a dramatic boost in accuracy.

G. Machine Learning Algorithms

So far, we have used Random Forests as learning algorithm in all our experiments. Nevertheless, there exist other algorithms that are well-suited for the considered problem. Therefore, we investigated the following set of well known machine learning algorithms with respect to their accuracy and robustness. For a detailed description of these algorithms we refer the interested reader to [5] and the references therein.

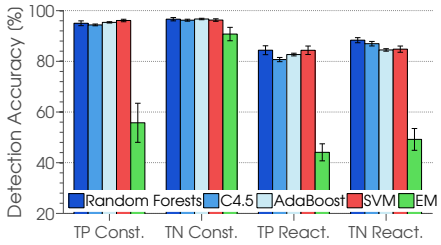
C4.5 Decision Tree: This algorithm relies on a single decision tree for classification. The input feature at each node of the tree is selected so as to maximize the information gain. Pruning is applied to reduce the size of the tree without degrading classification accuracy.

Adaptive Boosting (AdaBoost): AdaBoost iteratively combines multiple *weak* classifiers to obtain a single strong one. For this purpose, each individual classifier only needs to achieve a classification accuracy higher than 50%. Furthermore, the errors produced in one iteration are appropriately weighted in the next iteration. We select a maximum of 100 iterations and choose the C4.5 algorithm as *weak* classifier.

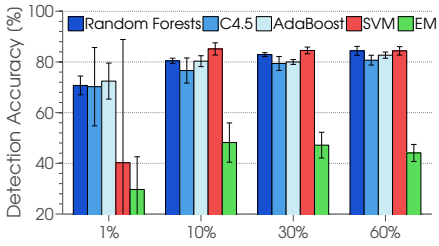
Support Vector Machine (SVM): This classifier looks for an hyperplane in a high dimensional space that maximizes the margin, i.e., the minimum distance between the hyperplane and a data point of any class. A non-linear transformation, by means of a kernel function, is applied to the data points to perform the classification in a higher dimensional space. We have used a gaussian kernel $k(x_i, x_j) = e^{-\gamma(x_i - x_j)}$ with $\gamma = 100$.

Expectation Maximization (EM): This learning algorithm is of unsupervised nature, hence, the class is not explicitly specified. The algorithm identifies patterns in the data points and groups them into clusters. We empirically determined that two clusters provide the best accuracy for outdoor training.

Results: Figure 9(a) shows the detection accuracy achieved by the different algorithms in the outdoor scenario. The four supervised learning algorithms exhibit a similar performance, although Random Forests and SVM achieve, on average, a marginally better detection. The EM algorithm yields a poor performance, which indicates that unsupervised learning is not



(a) Accuracy obtained outdoors with scenario-specific training.



(b) True positive (reactive jammer) accuracy obtained for different amounts of training data in the outdoor scenario.

Fig. 9. Comparison of the detection accuracy and robustness obtained by the different machine learning algorithms in the outdoor scenario.

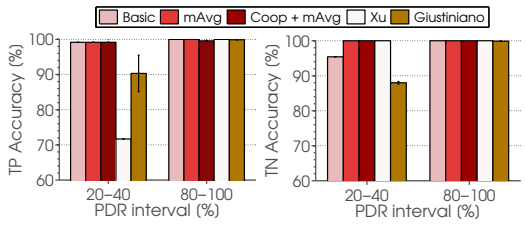
well-suited for the considered problem. Figure 9(b) shows the accuracy of the algorithms for a varying amount of training data. In general, all algorithms (expect for EM) provide an accurate detection already with 10% of the total training data. Random Forests and AdaBoost exhibit a very stable accuracy, which is in contrast with the large fluctuations experienced by SVM and C4.5, particularly with only 1% of the data.

H. Comparison with Related Work

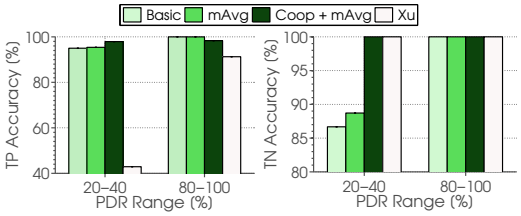
In literature there are only few works implementing a jamming detection scheme that could be applied (partially with significant modifications) in the context of 802.11 networks. We select the approaches presented in [7] and [24] and compare their accuracy against our scheme with respect to different link qualities, where the latter are characterized by the average PDR obtained while the jammer is silent. The accuracy of *Giustiniano's scheme* has been extracted from Figure 8 in [7][1]. In addition, we implement an approach similar to *Xu's method* [24]. For that, we generate a scatter plot containing samples of the PDR and (maximum) signal strength plus noise power collected without jammer activity. This is done similarly as in Figure 3, but for the data gathered in the indoor scenario described in Section IV-F. By inspecting the graph, we determine the operational *non-jammed* area. Later, any sample falling above that region is considered as a jamming attack. For more details we refer the reader to [24].

Figure 10 compares our approach in on-the-fly mode (basic design, moving average, and cooperation with moving average) against these two works for two different PDR ranges. We conducted the experiments in the indoor scenario described in Section IV-F. In general, we observe that all schemes (with punctual exceptions) are able to efficiently detect jamming

[1]Please note that the scenario, propagation conditions, and jammer behavior considered in the work may differ significantly from ours. Hence, this specific comparison should be treated with caution.



(a) Reactive jammer comparison.



(b) Constant Jammer comparison.

Fig. 10. Comparison of our approach with Xu [24] and Giustiniano [7].

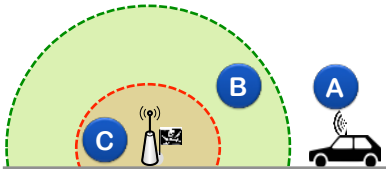
attacks when the PDR is larger than 80%. Under challenging propagation conditions, Xu's approach [24] yields a poor accuracy with respect to TP rates, while (for the reactive jammer) Giustiniano's approach [7] provides a better detection. Nevertheless, our scheme outperforms these two works significantly (for both jammers and link conditions). In cases where the basic design falls short in providing a successful detection (e.g., low TN rate for the constant jammer), the combination of cooperation and moving average achieves a remarkable performance.

I. Tuning Detection Sensitivity

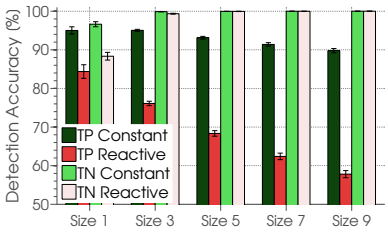
In general, we are interested in a timely and accurate jamming detection, particularly in the context of safety-critical applications over VANETs. Figure 11(a) illustrates the situation where a car moves towards a jammer. In this scenario, the communication conditions can be divided into three regions [17]. First, Region A is completely outside the interference range of the jammer. In Region B, the jammer impacts the communication but not enough to completely block it. Hence, safety-critical applications are expected to still work reliably. Finally, in Region C the vehicle is not able to successfully receive any packet. The dimensions of these regions depend on the transmit power of the devices, the network topology, and the jammer type, among others.

However, in this kind of scenarios the presence of a jammer is an event that can be expected to occur only sporadically. Assuming a higher probability of *unjammed* situations (such as in Region A), it is necessary to keep a very low rate of false positive detections. However, it is also desirable to have high true positive rates in Regions B and C. In the following, we propose and evaluate a method to address this issue.

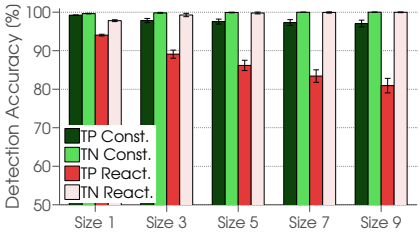
Figure 7(b) has shown that erroneous jamming detections (i.e., false positives) rarely occur in a consecutive manner. In the outdoor scenario, error bursts larger than 5 s happen in only 0.1% and in 0.4% of the cases for the constant and reactive jammer, respectively. Based on this observation, we propose a strategy to lower the false positive detections in



(a) Interference regions with different implications to the communication and to the jamming detection requirements as a vehicle approaches a jammed area.



(b) We apply an *absolute majority voting window* to limit the sensitivity to false positive events of our detection framework in the outdoor scenario.



(c) The combination of moving average (window of 5 s) with the absolute majority voting window achieves very high TN rates, while keeping acceptable TP rates for both jammers.

Fig. 11. Discussion on the applicability of our detection framework in vehicular scenarios.

unjammed situations (i.e., Region A). That strategy, which we refer to as *absolute majority vote* work as follows: A jammer is assumed to be present when *all* collected predictions within a specific time span give a positive answer. The corresponding results for different window sizes are shown in Figure 11(b). This method achieves TN rates that are very close to 100% for both jammers. For instance, in the case of 5 s window, the values are 99.96% and 99.95% for constant and reactive jammer, respectively. As expected, the sensitivity to jamming attacks happening in Regions B and C is degraded, particularly for the reactive jammer. This can be improved by combining this strategy with the moving average method presented in Section IV-E. As illustrated in Figure 11(c), the TP rates can be significantly improved. For instance, employing a moving average of 5 s yields a detection accuracy of 97% for the constant jammer and a TN rate of 99.92%. Finally, we believe that the appropriate cooperation between vehicles (e.g., by using additional context information such as GPS coordinates) will further improve the overall performance.

V. Discussion

In this work, we have considered two jamming attacks, namely reactive and constant. The jamming signals, as discussed in Section II, do not comply with the 802.11 standard nor do they exploit any knowledge about the protocol of the targeted network. We believe that a jammer emitting 802.11 compliant frames would not affect the metrics in a significantly different way than our jammer does. However, one major difference is expected with respect to the noise metric, since legitimate packets are not considered for that computation. This metric, however, provides only a modest improvement of the detection rate (cf. Figure 5).

A smarter jammer could, for instance, deliberately allow the successful reception of probing packets so as to tamper with the PDR computation. The jammer would need to distinguish the small probing packets from other packet types, which could be hindered by randomly adding padding bits to the packets.

Ideally, to provide accurate detection rates for a novel jammer type, our approach needs to gather training data that captures the impact of that specific jammer. In this context, we face two issues: First, collecting training data requires effort and the existence of the jammer is mandatory. Second, there is an indeterminably large number of jamming attacks that can be obtained by changing the interference signal pattern [11], [12] or adding protocol-awareness to the attacker [16], among others. As a result, we believe that there is no jamming detection strategy that guarantees the detection of all potential jammer types. To some extent the same problem is faced by computer anti-virus programs that need to regularly update their database with the fingerprints of new unseen viruses.

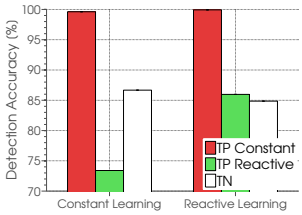


Fig. 12. Detection accuracy achieved exclusively with training data from a particular attack. Interestingly, using only reactive jammer training data for detecting a constant jammer achieves a successful detection. On the contrary, exclusively training with a constant jammer does not provide an accurate detection of the reactive jammer. These results were obtained in the indoor scenario introduced in Section IV-F.

Hence, our approach should not be considered as a one-fits-all solution that detects all possible jamming attacks on 802.11 networks. It is rather a methodology that improves the adaptability to novel jammers with low effort, while keeping a high detection accuracy. In this context, we made the interesting observation that training with a particular jammer can be reused to successfully detect other jamming attacks. This is illustrated in Figure 12, where the learning conducted in the presence of the reactive jammer is able to accurately detect constant jammer activity. We hence believe that training data that accounts for a representative amount of attacks, has the potential to accurately detect a wider range of jammers.

VI. Related Work

With the widespread deployment of wireless networks, especially 802.11-based WLANs, many research efforts have focused on jamming attacks due to their potential for compromising both reliability and security. Many works have characterized the impact of jamming on the network performance and discussed the reasons for the observed jamming effectiveness [4], [8]. Some other works have proposed methods to (partially) overcome the effects of jamming by using specific transmission technologies [14] or by appropriately tuning transmission parameters [15], among others.

In the cases where the robustness of the system to jamming cannot be increased, it is important to, at least, detect the pres-

ence of a jammer. Several jamming detection approaches for wireless networks have been proposed in the past years [24], [7], [20], [22], [10]. However, the majority of these works evaluate the proposed approaches only by means of simulations [20], [22] or not at all [10]. In [24] Xu et al. propose the use of measured energy together with the packet delivery ratio (PDR) for jamming detection in wireless sensor networks. The authors implement the approach in sensor devices and show that different jamming attacks can be identified. However, the approach is not directly transferrable to 802.11 networks as energy measurements as applied in Xu's work are not applicable with commodity hardware in 802.11.

Giustiniano et al. present in [7] an approach for detecting reactive jamming in direct sequence spread spectrum (DSSS) wireless systems (e.g., 802.11b/g). The authors characterize the relationship between the chip error rate measured over the preamble (where the reactive jammer is assumed to be silent) and the actual frame error rate. During operation, transmission events that diverge from the previously characterized behavior are assumed to be caused by a reactive jamming signal. The authors implement and evaluate their approach on an USRP platform. They measure a detection rate with a false negative rate below 5% under good channel conditions, while the accuracy decreases under challenging conditions (e.g., the false negative rate rises up to 30% for links with a PDR below 25%). The approach has a limited applicability, as it is only useful to detect reactive jamming in DSSS-based systems. This is, however, not the common case in 802.11, where OFDM is the de-facto PHY present in current and considered for future WLAN generations. Furthermore, the proposed metric is not provided by commodity 802.11 hardware.

VII. Conclusions

In this paper, we have presented a machine learning-based jamming detection approach for 802.11 networks that works with commodity off-the-shelf hardware. We have experimentally evaluated our approach and showed that it achieves an extraordinarily high accuracy both for true positives and negatives in indoor and mobile outdoor scenarios, under different propagation conditions (good- and bad-links, with and without concurrent traffic from neighbor networks), and for constant and reactive jammer types. Although our approach is a standalone tool that does not rely on other applications or information from other nodes in the network, we have incorporated a cooperative approach that can be enabled on demand. We have shown that exploiting the knowledge of past predictions in combination with cooperative jamming detection significantly improves the detection accuracy introducing only low overhead. Furthermore, we have compared different popular machine learning algorithms with respect to their accuracy and robustness. Finally, we have shown by means of measurements that our approach outperforms related work significantly, especially in scenarios with poor link conditions.

VIII. Acknowledgments

This research was funded in part by the DFG Cluster of Excellence on Ultra High-Speed Mobile Information and Communication (UMIC).

References

[1] Ath9k - Linux Wireless: Official Website. http://wireless.kernel.org/en/users/Drivers/ath9k. Last visit 19-12-2013.

[2] Method and System for Noise Floor Calibration and Receive Signal Strength Detection (Atheros Patent). http://www.patentstorm.us/patents/7245893/description.html. Last visit 19-12-2013.

[3] I. Aktas, F. Schmidt, M. Alizai, T. Drüner, and K. Wehrle. CRAWLER: An Experimentation Platform for System Monitoring and Cross-Layer-Coordination. In *Proc. IEEE WoWMoM*, 2012.

[4] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa. On the Performance of IEEE 802.11 under Jamming. In *Proc. IEEE INFOCOM*, 2008.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[6] L. Breiman. Random Forests. Technical report, 2001.

[7] D. Giustiniano, V. Lenders, J. Schmitt, M. Spuhler, and M. Wilhelm. Detection of Reactive Jamming in DSSS-based Wireless Networks. In *Proc. ACM WiSec*, 2013.

[8] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and Mitigating the Impact of RF Interference on 802.11 Networks. In *Proc. ACM SIGCOMM*, 2007.

[9] M. Hamalainen, V. Hovinen, R. Tesi, J. H. Iinatti, and M. Latva-aho. On the UWB System Coexistence with GSM900, UMTS/WCDMA, and GPS. *IEEE Journal on Selected Areas in Communications*, 2002.

[10] A. Hamieh, J. Ben-Othman, and L. Mokdad. Detection of Radio Interference Attacks in VANET. In *Proc. of GLOBECOM*, 2009.

[11] I. Harjula, J. Pinola, and J. Prokkola. Performance of IEEE 802.11 based WLAN Devices under Various Jamming Signals. In *Proc. IEEE MILCOM*, 2011.

[12] T. Karhima, A. Silvennoinen, M. Hall, and S.-G. Haggman. IEEE 802.11b/g WLAN tolerance to jamming. In *Proc. IEEE MILCOM*, 2004.

[13] A. Khattab, J. Camp, C. Hunter, P. Murphy, A. Sabharwal, and E. W. Knightly. WARP: A Flexible Platform for Clean-Slate Wireless Medium Access Protocol Design. *ACM SIGMOBILE Mobile Computing and Communications Review*, 2008.

[14] V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein. Using Channel Hopping to Increase 802.11 Resilience to Jamming Attacks. In *Proc. IEEE INFOCOM*, 2007.

[15] K. Pelechrinis, I. Broustis, S. Krishnamurthy, and C. Gkantsidis. A Measurement-Driven Anti-Jamming System for 802.11 Networks. *IEEE/ACM Transactions on Networking*, 2011.

[16] K. Pelechrinis, M. Iliofotou, and S. Krishnamurthy. Denial of Service Attacks in Wireless Networks: The Case of Jammers. *IEEE Communications Surveys Tutorials*, 13(2):245 –257, May 2011.

[17] O. Puñal, A. Aguiar, and J. Gross. In VANETs We Trust?: Characterizing RF Jamming in Vehicular Networks. In *Proc. ACM VANET*, 2012.

[18] C. Shahriar, S. Sodagari, and T. C. Clancy. Physical-Layer Security Challenges of DSA-Enabled TD-LTE. In *Proc. ACM CogART*, 2011.

[19] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. MOJO: A Distributed Physical Layer Anomaly Detection System for 802.11 WLANs. In *Proc. ACM MobiSys*, 2006.

[20] G. Thamilarasu, S. Mishra, and R. Sridhar. Improving Reliability of Jamming Attack Detection in Adhoc Networks. *IJCNIS*, 2011.

[21] I. Tinnirello, D. Giustiniano, L. Scalia, and G. Bianchi. On the Side-Effects of Proprietary Solutions for Fading and Interference Mitigation in IEEE 802.11b/g Outdoor Links. *Computer Networks*, 2009.

[22] A. L. Toledo and X. Wang. Robust Detection of MAC Layer Denial-of-Service Attacks in CSMA/CA Wireless Networks. *Transactions on Information Forensics and Security*, 2008.

[23] D. Xu and R. Bagrodia. JamDetect: A System to Detect RAA Aware Jamming Attacks in IEEE 802.11 Networks. In *Proc. IEEE MILCOM*, 2012.

[24] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In *Proc. ACM MobiHoc*, 2005.