A Measurement-Driven Anti-Jamming System for 802.11 Networks

Konstantinos Pelechrinis, *Member, IEEE*, Ioannis Broustis, *Member, IEEE*, Srikanth V. Krishnamurthy, *Senior Member, IEEE*, and Christos Gkantsidis, *Member, IEEE, ACM*

***Abstract*—Dense, unmanaged IEEE 802.11 deployments tempt saboteurs into launching jamming attacks by injecting malicious interference. Nowadays, jammers can be portable devices that transmit intermittently at low power in order to conserve energy. In this paper, we first conduct extensive experiments on an indoor 802.11 network to assess the ability of two physical-layer functions, rate adaptation and power control, in mitigating jamming. In the presence of a jammer, we find that: 1) the use of popular rate adaptation algorithms can significantly degrade network performance; and 2) appropriate tuning of the carrier sensing threshold allows a transmitter to send packets even when being jammed and enables a receiver to *capture* the desired signal. Based on our findings, we build ARES, an Anti-jamming REinforcement System, which tunes the parameters of rate adaptation and power control to improve the performance in the presence of jammers. ARES ensures that operations under benign conditions are unaffected. To demonstrate the effectiveness and generality of ARES, we evaluate it in three wireless test-beds: 1) an 802.11n WLAN with MIMO nodes; 2) an 802.11a/g mesh network with mobile jammers; and 3) an 802.11a WLAN with TCP traffic. We observe that ARES improves the network throughput across all test-beds by up to 150%.**

***Index Terms*—IEEE 802.11, jamming, power control, rate control.**

I. Introduction

THE WIDESPREAD proliferation of IEEE 802.11 wireless networks makes them an attractive target for saboteurs with jamming devices [1]–[4]. This makes the defense against such attacks very critical. A jammer transmits electromagnetic energy to hinder legitimate communications on the wireless medium. A jamming attack can cause the following effects in an 802.11 network: 1) due to carrier sensing, cochannel transmitters defer their packet transmissions for prolonged periods; and 2) the jamming signal collides with legitimate packets at receivers. Frequency-hopping techniques have been previously proposed for avoiding jammers [5], [6]. Such schemes, however, are not effective in scenarios with wideband jammers [7], [8]. Furthermore, given that 802.11 operates on relatively few frequency channels, multiple jamming devices operating on different channels can significantly hurt performance in spite of using frequency hopping [9].

In this paper, we ask the following question: *How can legacy 802.11 devices alleviate the effects of a jammer that resides on the same channel used by a legitimate communicating pair in real time?* We address this challenge by developing ARES[1] (Anti-jamming REinforcement System), a novel measurement-driven system, which detects the presence of jammers and invokes rate adaptation and power control strategies to alleviate jamming effects. Clearly, not much can be done to mitigate jammers with unlimited resources in terms of transmission power and spectrum efficiency. Note, however, that in a plurality of cases the jamming device can be resource-constrained, with capabilities similar to that of the legitimate device.[2] Portable, battery-operated jammers are typically configured to transmit intermittently and sometimes at low power in order to conserve energy and harm the network for extended periods of time. In addition, misconfiguration of "legitimate" devices can transform them to resource-constrained jammers [3]. In such cases, ARES can effectively *fight* against the malicious entity, as we discuss later. Our contributions are the following.

1) *Understanding the impact of jammers in an 802.11 network with rate/power control.* First, we perform an in-depth measurement-based experimental study on our indoor test-bed to quantify the impact of jamming when employing rate and/or power control. To the best of our knowledge, there are no such studies to date. With rate control, a transmitter can increase or lower its transmission rate depending on the observed packet delivery ratio (PDR) at the receiver. With power control, nodes may increase their transmission powers and/or clear channel assessment (CCA) thresholds [10] in order to increase the probability of successful packet reception. The design of ARES is driven by two key experimental observations.
   a) *Rate adaptation can be counterproductive.* In the presence of a jammer that is active intermittently (and sleeps in between), the use of rate adaptation is not always beneficial. We conduct experiments with three popular rate adaptation algorithms: SampleRate [11], Onoe [12], and Adaptive Multi-Rate Retry (AMRR) [13]. With every scheme, we observe that the use of rate adaptation may work

Manuscript received February 02, 2010; revised September 30, 2010; accepted December 20, 2010; approved by IEEE/ACM Transactions on Networking Editor D. Agrawal. Date of publication February 04, 2011; date of current version August 17, 2011.

K. Pelechrinis is with the University of Pittsburgh, Pittsburgh, PA 15260 USA (e-mail: kpele@pitt.edu).

I. Broustis and S. V. Krishnamurthy are with the University of California, Riverside, CA 92521 USA (e-mail: broustis@cs.ucr.edu; krish@cs.ucr.edu).

C. Gkantsidis is with Microsoft Research Cambridge, Cambridge CB3 0FB, U.K. (e-mail: chrisgk@microsoft.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2011.2106139

[1] ARES [pron. "áris"] was the god of war in Greek mythology. We choose the name as a symbol of the combat with jammers.

[2] We implement a jamming utility on a commodity 802.11 network interface card (NIC) as described in more detail in Section III.

in favor of the jammer. This is because, rate adaptation wastes a large portion of a jammer's sleeping time in order to gradually converge to the "best" rate. We analytically determine when fixed rate operations may be preferable to the use of rate adaptation.

b) *Tuning the carrier sense threshold is beneficial.* We collect throughput measurements with many different transmission powers and CCA thresholds. We find the following: 1) In the presence of a jammer, legitimate transmissions with maximum power could lead to significant benefits only when operating at low data rates. 2) Increasing the CCA threshold can allow a transmitter that is being jammed to send packets and, in addition, facilitate the *capture* of packets in the presence of jamming interference. Together, these effects can significantly reduce the throughput degradation.

2) *Designing ARES, a novel anti-jamming system.* The above observations drive the design of ARES. ARES primarily consists of two modules. The *rate control module* chooses between fixed-rate assignment and rate adaptation, based on channel conditions and the jammer characteristics. The primary objective of this module is to effectively utilize the periods when a jammer is asleep. The *power control module* adjusts the CCA threshold to facilitate the transmission and the reception (*capture*) of legitimate packets during jamming. Care is taken to avoid starvation of nodes due to the creation of asymmetric links [10]. This module is used to facilitate successful communications while the jammer is active. Although rate and power control have been proposed as interference alleviation techniques, their behavior has not been studied in jamming environments. To our knowledge, our work is the first to conduct such a study.

3) *Implementing and experimentally validating ARES.* We implement and evaluate the modules of ARES on real hardware, thereby making ARES one of the few anti-jamming system implementations for 802.11 networks. ARES relies on the existence of an accurate jamming detection module. It is beyond the scope of our work to design a new detection scheme, and thus we incorporate a mechanism proposed previously in [14]. To demonstrate the effectiveness and generality of our system, we apply it on three different experimental networks: an 802.11n WLAN with multiple-input–multiple-output (MIMO)-enabled nodes, an 802.11a/g mesh network with mobile jammers, and a static 802.11a WLAN with uplink TCP traffic. Our measurements demonstrate that ARES provides performance benefits in all the three networks. Throughput improvements of up to 150% are observed.

The remainder of this paper is structured as follows. In Section II, we provide some background on jamming and discuss related studies. In Section III, we describe our wireless test-bed and the experimental methodology. We describe our extensive experiments to understand the impact of rate and power control in the presence of a jammer in Sections IV and V, respectively. In Section VI, we construct ARES based on our observations. We present our evaluations of ARES in Section VII. Section VIII discusses the scope of our study. We conclude in Section IX.

II. Background and Related Work

In this section, first we briefly describe the operations of a jammer and its attack capabilities. Next, we discuss relevant previous studies.

A. Types of Jamming Attacks

Jammers can be distinguished in terms of their attack strategy. A detailed discussion can be found in [14].

1) Nonstop Jamming: Constant jammers continuously emit electromagnetic energy on a channel. Nowadays, constant jammers are commercially available and easy to obtain [1], [7]. While constant jammers emit nondecipherable messages, *deceptive* jammers transmit seemingly legitimate back-to-back dummy data packets. Hence, they can mislead other nodes and monitoring systems into believing that legitimate traffic is being sent.

2) Intermittent Jamming: As the name suggests, these jammers are active intermittently; the primary goal is to conserve battery life. A *random* jammer typically alternates between uniformly distributed jamming and sleeping periods. It jams for T_{j} s, and then it sleeps for T_{s} s. A *reactive* jammer starts emitting energy only if it detects traffic on the medium. This makes the jammer difficult to detect. However, implementing reactive jammers can be a challenge.

For the purposes of this paper, we primarily consider the deceptive-random jammer model. Attackers are motivated into using a random jammer because putting the jammer to sleep intermittently can increase its lifetime and decrease the probability of detection [14]. Furthermore, it is the most generalized representation of a jammer. Appropriately choosing the sleep times could turn the jammer into a constant jammer or (with high probability) a reactive jammer. Moreover, reactive jammers are not easily available since they are harder to implement and require special expertise on the part of the attacker. Nevertheless, we show the applicability of ARES with constant and reactive jammers in Section VIII.

B. Related Work

Most previous studies employ frequency hopping to avoid jammers. Frequency hopping, however, cannot alleviate the influence of a wideband jammer [7], [8], which can effectively jam all the available channels. In addition, recent studies have shown that a few cleverly coordinated, narrowband jammers can practically block the whole spectrum [9]. Thus, ARES does not rely on frequency hopping.

1) Studies Based on Frequency Hopping: Navda *et al.* [5] implement a proactive frequency-hopping protocol with pseudorandom channel switching. They compute the optimal frequency-hopping parameters, assuming that the jammer is aware of the procedure followed. Xu *et al.* [6] propose two anti-jamming techniques: reactive channel surfing and spatial retreats. However, their work is on sensor networks that only support very low data rates and transmission powers. Gummadi *et al.* [15] find that 802.11 devices are vulnerable to specific patterns of narrowband interference related to time recovery, dynamic range selection, and PLCP-header processing. They show that due to these limitations, an intelligent jammer with a $1000\times$ weaker signal (than that of the legitimate transceiver) can still corrupt the reception of packets. In order to alleviate these effects, they propose a rapid frequency-hopping strategy.

2) Other Relevant Work: Xu *et al.* [14] develop efficient mechanisms for jammer detection at the PHY layer (for all the four types of jammers). However, they do not propose any jamming mitigation mechanisms. In [16], the same authors suggest that competition strategies, where transceivers adjust their transmission powers and/or error correction codes, *might* alleviate jamming effects. However, they neither propose an anti-jamming protocol nor perform evaluations to validate their suggestions. Lin and Noubir [17] present an analytical evaluation of the use of cryptographic interleavers with different coding schemes to improve the robustness of wireless LANs. In [18], the authors show that in the absence of error-correction codes (as with 802.11) the jammer can conserve battery power by destroying only a portion of a legitimate packet. Noubir [19] also proposes the use of a combination of directional antennas and node mobility in order to alleviate jammers. ARES can easily be used in conjunction with directional antennas or with error correction codes. We would like to refer the interested reader to our literature survey on anti-jamming systems in [20] for more details.

3) Prior Work on Rate and Power Control: Rate and power control techniques have been proposed in the literature as means of mitigating interference (e.g., [10], [11], [21]–[23], and the references therein). However, they do not account for a hostile jamming environment. Wth these schemes, nodes cooperate in order to mitigate the impact of "legitimate" interference, thereby improving the performance. As an example, Zhai and Fang [23] consider the optimal carrier sensing range for maximum spatial reuse in MANETs. All nodes are restricted to the same maximum transmission power, and their work is purely based on analysis and simulations. In this paper, we follow a purely experimental approach, and our results indicate that ARES effectively alleviates the impact of jammers that use higher transmission powers. Our scheme is specialized toward handling malicious interference of jammers, which attempt to disrupt ongoing communications.

III. Experimental Setup

In this section, we describe our wireless test-bed and the experimental methodology that we follow.

A. Test-Bed Description

Our wireless test-bed [24] is deployed on the third floor of Engineering Building II at the University of California, Riverside. Our test-bed consists of 37 Soekris net4826 nodes [25], which mount a Debian Linux distribution with kernel v2.6, over NFS. The node layout is depicted in Fig. 1. Thirty of these nodes are each equipped with two miniPCI 802.11a/g WiFi cards, an *EMP-8602 6 G* with Atheros chipset, and an *Intel-2915*. The other seven nodes are equipped with one *EMP-8602 6 G* and one *RT2860* card that supports MIMO-based (802.11n) communications. We use the MadWifi driver [26] for the *EMP-8602 6 G* cards. We have modified the Linux client driver [27] of the *RT2860* to enable Space Time Block Coding (STBC) support. We use a proprietary version of the *ipw2200* access point (AP) and client driver/firmware of the *Intel-2915* card. With this version, we are able to tune the CCA threshold parameter.

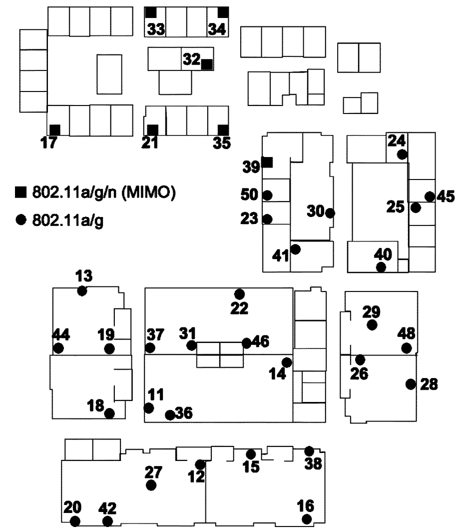


Fig. 1. Deployment of our wireless test-bed.

B. Experimental Settings and Methodology

We experiment with different rate adaptation algorithms in the presence of random jammers. We also perform experiments with various transmission powers of jammers and powers/CCA thresholds of legitimate nodes. Our measurements encompass an exhaustive set of wireless links, routes of different lengths, as well as static and mobile jammers. We examine both single-input–single-output (SISO) and MIMO links. We experiment with three modes of operation: 802.11a/g/n (unless otherwise stated throughout this paper, our observations are consistent for all three modes of operation). The experiments are performed late at night in order to isolate the impact of the jammers by avoiding interference from colocated WLANs. By default, all devices (legitimate nodes and jammers) set their transmission powers to 18 dBm (for our experiments that involve only the Intel-2915 cards, the maximum power that we can use is 20 dBm).

1) Implementing a Random Jammer: Our implementation of a random jammer is based on a specific configuration ($\mathrm{CCA} = 0$ dBm) and a user space utility that sends broadcast packets as fast as possible. For the purposes of research, we have implemented our own random jammer on an 802.11 legacy device by setting the CCA threshold to 0 dBm. By setting the CCA threshold to such a high value, we force the device to ignore all legitimate 802.11 signals even after carrier sensing. Packets arrive at the jammer's circuitry with powers less than 0 dBm (even if the distances between the jammer and the legitimate transceivers are very small). An effective random jammer should be able to transmit packets on the medium, as fast as possible, during random *active* time intervals. We develop a user-space software utility with the following functionalities.

- The jammer transmits broadcast UDP traffic. This ensures that its packets are transmitted back to back and that the jammer does not wait for any ACK messages (by default, the backoff functionality is disabled in 802.11 for broadcast traffic). In other words, this setup allows the jamming node to defer its back-to-back transmissions for the minimum possible time (i.e., $\mathrm{DIFS} + \min_{\mathrm{BackOff}}$). Our utility employs *raw sockets*, which allow the construction of UDP

packets from scratch and the forwarding of each packet directly down to the hardware. Note that this implementation *bypasses the 802.11 protocol*, and hence, the jammer does not wait in the backoff state after each packet transmission.
- Our utility schedules uniformly distributed random jamming intervals. The jammer is in the active state for a random period of time, during which it constantly transmits packets back to back. It then transits to an idle (sleeping) state for a different, randomly chosen period of time during which it does not emit energy. The two states alternate, and their durations are computed anew at the beginning of each cycle (a cycle consists of an active and an idle period).

We use a set of four nodes as jammers on our test-bed. These are equipped with *Intel-2915* cards that allow CCA tuning.

2) Traffic Characteristics: We utilize the *iperf* measurement tool to generate UDP data traffic among legitimate nodes; the packet size is 1500 B. The duration of each experiment is 1 h. For each experiment, we first enable *iperf* traffic between legitimate nodes, and subsequently, we activate the jammer(s). We consider both mesh and WLAN connectivity. We experiment with different jammer distributions, namely: 1) *frequent jammers*, which are active almost all of the time; 2) *rare jammers*, which spend most of their time sleeping; and 3) *balanced jammers* that have similar average jamming and sleeping times. We have disabled RTS/CTS message exchange throughout our experiments (a common design decision in practice [28]).

IV. System Guidelines for Rate Adaptation

Rate adaptation algorithms are utilized to select an appropriate transmission rate as per the current channel conditions. As interference levels increase, lower data rates are dynamically chosen. Since legitimate nodes consider jammers as interferers, rate adaptation will reduce the transmission rate on legitimate links while jammers are active. Hence, one could potentially argue that rate control on legitimate links increases reliability by reducing rate and can thus provide throughput benefits in jamming environments.

To examine the validity of this argument, we experiment with three different popular rate adaptation algorithms, SampleRate [11], AMRR [13], and Onoe [12]. These algorithms are already implemented on the MadWifi driver that we use. For simplicity, we first consider a balanced random jammer, which selects the sleep duration from a uniform distribution $U[1,8]$ and the jamming duration from $U[1,5]$ (in seconds).

A. Details on the Experimental Process

We perform experiments with both single-hop and multihop configurations. In each experiment, we first load the particular rate-control Linux-kernel module (SampleRate, AMRR or Onoe) on the wireless cards of legitimate nodes. We initiate data traffic between the nodes and activate the jammer after a random time. We collect throughput measurements on each data link once every 500 ms. We use the following terminology.

1) *Fixed transmission rate* $R_{\rm f}$*:* This is the nominal transmission rate configured on the wireless card.
2) *Saturated rate* $R_{\rm s}$*:* It is the rate achieved when $R_{\rm f}$ is chosen to be the rate on the wireless card. In order to compute $R_{\rm s}$, for a given $R_{\rm f}$, we consider links where the PDR is 100% for the particular setting of $R_{\rm f}$. We then measure the rate achieved in practice. We notice that for lower values of $R_{\rm f}$, the specified rate is actually achieved on such links. However, for higher values of $R_{\rm f}$ (as an example, $R_{\rm f}$ = 54 Mb/s), the achieved data rate is much lower; this has been observed in other work, e.g., [29]. Table I contains a mapping, derived from measurements on our test-bed, between $R_{\rm f}$ and $R_{\rm s}$.
3) *Application data rate* $R_{\rm a}$*:* This is the rate at which the application generates data.

TABLE I
Saturated Throughput Matrix in Megabits per Second

R_f	6	9	12	18	24	36	48	54
R_s	6	9	12	18	24	26	27	27

It is difficult (if not impossible) to *a priori* determine the *best* fixed rate on a link. Given this, and if we let R be the set of all possible fixed transmission rates, we set

$$R_{\rm f} = \{\min_{x \in R} x : x \geq R_{\rm a}\}$$

which is the maximum rate that is required by the application (we discuss the implications of this choice later). Our key observations are summarized as follows.

- Rate adaptation algorithms perform poorly on high-quality links due to the long times that they incur for converging to the appropriate high rate.
- On *lossless* links, the fixed rate $R_{\rm f}$ is better, while rate adaptation is beneficial on *lossy* links.

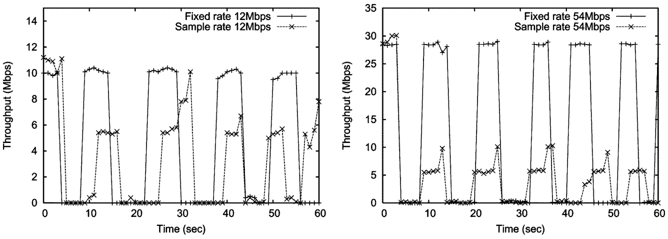
We defer defining what constitute lossless or lossy links later in this section. Conceptually, we consider lossless links to be those links that can achieve higher long-term throughput using a fixed transmission rate $R_{\rm f}$ rather than by applying rate adaptation.

B. Single-Hop Configurations

Our experiments with one-hop connectivity involve 80 sets of sender–receiver pairs and one jammer per pair. We impose that a jammer interferes with *one* link at a time and that the legitimate data links do not interfere with each other. Thus, we perform 20 different sets of experiments, with four isolated data links and four jammers in each experiment.

1) Rate Adaptation Consumes a Significant Part of the Jammer's Sleep Time to Converge to the Appropriate Rate: As soon as the jammer "goes to sleep," the link quality improves, and thus the rate control algorithm starts increasing the rate progressively. However, since the purpose of a jamming attack is to corrupt as many transmissions as possible, the jammer will typically not sleep for a long time. In such a case, the sleep duration of the jammer will not be enough for the rate control to reach the highest rate possible. To illustrate this, we choose two links on our test-bed, one that can support 12 Mb/s and the other that can support 54 Mb/s. Fig. 2 depicts the results. We observe the following: 1) irrespective of whether SampleRate or a fixed rate strategy is used, during jamming, the throughput drops to values close to zero since the jammer blocks the medium for the sender; and 2) *the throughput achieved with SampleRate is quite low, and much lower than if we fix the rate to the constant value of 12 Mb/s.* Note that we have observed the same behavior with AMRR and Onoe.

2) Fixed Rate Assignment Outperforms Rate Adaptation on Lossless Links: As was alluded to, in order to find the *best* rate



(a) (b)

Fig. 2. Rate adaptation algorithms may not find the best rate during the sleep period of the jammer. We show cases for (a) $R_{\rm a} = 12$ Mb/s and (b) $R_{\rm a} = 54$ Mb/s.

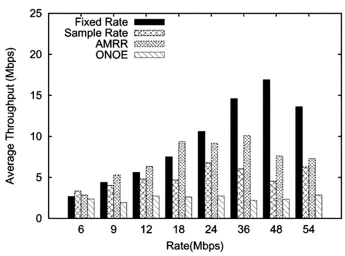


Fig. 3. Fixed rates outperform rate adaptation for high-quality links under random jamming. $(R_{\rm a} = R_{\rm f})$.

on a link after a period where there is no throughput due to a jammer, the rate adaptation mechanisms gradually increase the rate, invoking transmissions at all the lower rates interim, until the best rate is reached. For links that can inherently support high rates, this process might consume the sleep period of the jammer (as suggested by the results in Fig. 2). If the best rate for a link was known *a priori*, at the instance that the jammer goes to sleep, transmissions may be invoked at that rate. This would utilize the sleep period of the jammer more effectively. As observed in Fig. 3, the throughputs achieved with fixed rate assignment are much higher than those achieved with rate adaptation on such links.

C. Determining the Right Transmission Rate Policy

1) Implications of Setting $R_{\rm f} = \{\min_{x \in R}\ x : x \geq R_{\rm a}\}$*:* Since the application does not require the link to sustain a higher rate, the highest throughput for that application rate is reached either with this choice of $R_{\rm f}$ or with some rate that is lower than $R_{\rm a}$. If the rate adaptation algorithm converges to a rate that results in a throughput that is higher than with the chosen $R_{\rm f}$, then the adaptive rate strategy should be used. If instead, during the jammer's sleep period, the rate adaptation technique is unable to converge to such a rate, the fixed rate strategy is better.

2) Analytically Determining the Right Rate: In order to determine whether it is better to use a fixed- or an adaptive-rate approach for a given link, we perform an analysis based on the following parameters:

1) The distribution of the jammer's active and sleep periods (we call this the *jammer's distribution*).
2) The application data rate $R_{\rm a}$.
3) The performance metric on the considered legitimate link, i.e., PDR, link throughput, etc.
4) The rate adaptation scheme that is employed, i.e., Onoe, SampleRate, etc. The key scheme-specific factor is the transition time from a lower rate to the next higher rate under conducive conditions.
5) The *effectiveness* of the jammer F, measured by the achievable throughput while the jammer is on. The lower the throughput, the more effective the jammer.

Let us suppose that the expected *sleeping* duration of the jammer during a cycle is given by $E[t_{\rm s}]$, and the expected period for which it is active by $E[t_{\rm j}]$. The expected duration of a *cycle* is then $E[t_{\rm s}] + E[t_{\rm j}]$. As an example, if the jammer picks its sleeping period from a uniform distribution $U[a, b]$ and its jamming period from $U[c, d]$, $E[t_{\rm s}]$ and $E[t_{\rm j}]$ are equal to $\frac{b+a}{2}$ and $\frac{d+c}{2}$, respectively. For simplicity, let us assume that the link-quality metric employed[3] is the PDR. With application data rate $R_{\rm a}$ and *fixed* transmission rate $R_{\rm f}$, the throughput achieved during a jammer's cycle is

$$T_{\rm fixed} = \frac{E[t_{\rm s}]}{E[t_{\rm s}] + E[t_{\rm j}]} \cdot {\rm PDR_f} \cdot R_{\rm s} + \frac{E[t_{\rm j}]}{E[t_{\rm s}] + E[t_{\rm j}]} \cdot F \quad (1)$$

where ${\rm PDR_f}$ is the PDR of the link at rate $R_{\rm f}$. Recall that the rate achieved in practice with a specified rate $R_{\rm f}$ is $R_{\rm s}$.

To compute the throughput with *rate adaptation*, we proceed as follows. Let us assume that $x(F, R_{\rm s})$ corresponds to the convergence time of the rate adaptation algorithm (specific to the chosen algorithm). We consider the following two cases.

1) $x(F, R_{\rm s}) < E[t_{\rm s}]$. This case holds when the jammer's sleep duration is sufficient (on average) for the rate control algorithm to converge to the best rate $R_{\rm s}$. In this scenario, the achievable throughput is

$$T_{\rm adapt} = \frac{[E[t_{\rm s}] - x(R_{\rm s})] \cdot R_{\rm s} + \sum_{R_{\rm i}} y(R_{\rm i}) \cdot R_{\rm i} + E[t_{\rm j}] \cdot F}{E[t_{\rm s}] + E[t_{\rm j}]}$$

where $R_{\rm i} \in S$, S being the set of all intermediate rates from F to $R_{\rm s}$. $y(R_{\rm i})$ is the time that the rate control algorithm spends at the corresponding rate $R_{\rm i}$. The values of $y(R_{\rm i})$ are specific to the implementation of the rate control algorithm. Note that $x(F, R_{\rm s})$ can be easily computed from $y(R_{\rm i})$ by adding all the individual durations for the rates belonging to the set S.

2) $x(F, R_{\rm s}) \geq E[t_{\rm s}]$. In this scenario, the average sleep time of the jammer is insufficient for the rate control algorithm to converge to the desired rate. When the jammer wakes up, the rate will again drop due to increased interference. Here, the throughput that can be achieved during a jammer's cycle is

$$T_{\rm adapt} = \frac{\sum_{i=1}^{n} y(R_{\rm i}) \cdot R_{\rm i} + \left[E[t_{\rm s}] - \sum_{i=1}^{n} y(R_{\rm i})\right] \cdot R_{n+1} + E[t_{\rm j}] \cdot F}{E[t_{\rm s}] + E[t_{\rm j}]}$$

where $n = \max\left\{k : \sum_{i=1}^{k} y(R_{\rm i}) \leq E[t_{\rm s}]\right\}$.

Based on the above analysis, we define a link to be *lossy*, when $T_{\rm fixed} \leq T_{\rm adapt}$. The links on which $T_{\rm fixed} > T_{\rm adapt}$ are classified as *lossless* links. Clearly, for lossy links it is better to use the rate adaptation algorithm. The analysis can be used to compute ${\rm PDR_f^{TH}}$, a threshold value of ${\rm PDR_f}$ below which a

[3]Our analysis can be modified to adopt any other link-quality metric.

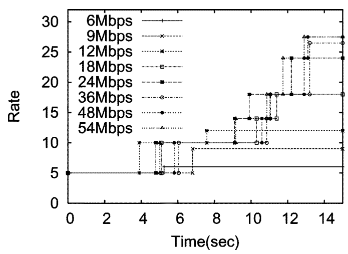


Fig. 4. Measured convergence times of the MadWifi SampleRate algorithm for the different application data rates.

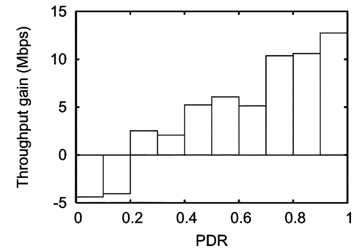


Fig. 5. Throughput gain of fixed rate versus SampleRate, for various link qualities and for application data rate of 54 Mb/s.

TABLE II
$\mathrm{PDR_f}$ THRESHOLDS

R_f	Measured PDR_f^{TH}	Analytical PDR_f^{TH}
6	0.82	0.83
9	0.52	0.55
12	0.40	0.41
18	0.26	0.27
24	0.19	0.21
36	0.19	0.20
48	0.17	0.185
54	0.15	0.185

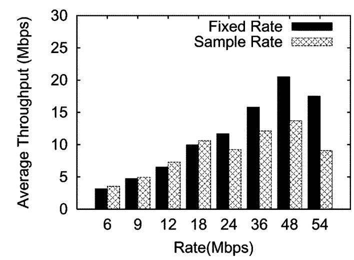


Fig. 6. Performance with rare jammers is aligned with our observations for the case with balanced jammers. $(R_\mathrm{a} = R_\mathrm{f})$.

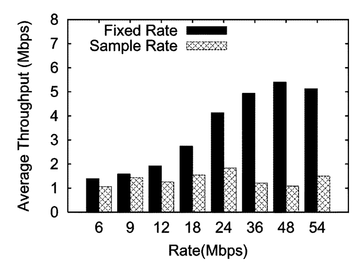


Fig. 7. Fixed rate improves the performance more than rate adaptation at high rates with frequent jammers. $(R_\mathrm{a} = R_\mathrm{f})$.

rate adaptation strategy performs better than the fixed rate approach. In particular, by setting $T_\mathrm{fixed} = T_\mathrm{adapt}$ and solving this equation, one can compute $\mathrm{PDR_f^{TH}}$. Based on this, a decision can be made on whether to enable rate adaptation or use fixed-rate assignment. If the observed PDR is larger than the computed threshold, fixed rate should be used. Otherwise, rate adaptation should be used.[4]

D. Validation of Our Analysis

In order to validate our analysis, we measure $\mathrm{PDR_f^{TH}}$ on 80 different links in the presence of a balanced jammer. We then compare them against the $\mathrm{PDR_f^{TH}}$ values computed with our analysis. Note here that the analysis itself depends on measured values of certain quantities (such as the jammer distribution and the function $y(R_\mathrm{i})$). In this experiment, we consider the SampleRate algorithm and measure the values of $x(F, R_\mathrm{s})$ and $y(R_\mathrm{i})$. The jammer's sleep time follows $U[0, 4]$, and the jamming time follows $U[1, 6]$. Fig. 4 plots the values of function y for different values of R_f.

In Table II, we compare the theoretically computed PDR thresholds to the ones measured on our test-bed for various values of R_f. We observe that the $\mathrm{PDR_f}$ thresholds computed with our analysis are very similar to the ones measured on our test-bed. There are slight discrepancies since our analysis is based on using measured average values that may change to some extent over time. We wish to stress that while we verify our analysis assuming that the jammer is active and idle for uniformly distributed periods of time, our analysis depends only on expected values and is therefore valid for other jammer distributions. Finally, Fig. 5 shows the advantage of using a fixed rate approach over SampleRate for various PDR values and with $R_\mathrm{f} = 54$ Mbps. We observe that SampleRate provides higher throughput only for very low PDR values.

[4]Note here that if the jammer temporally varies the jamming and timing cycles, our analysis is still valid, but for shorter timescales (cycles over which these distributions are stationary).

TABLE III
JAMMING DISTRIBUTIONS THAT WE USE IN OUR EXPERIMENTS

-	Sleep time (sec)	Jamming time (sec)
Balanced	U[1,8]	U[1,5]
Rare	U[1,5]	U[1,2]
Frequent	U[1,2]	U[1,15]

Next, we consider two extreme cases of jamming: frequent and rare jammers (see Section III). The distributions that we use in our experiments for these jammers are shown in Table III. Note that by choosing the jammer's sleeping and jamming time from distributions like that of the frequent jammer, we essentially construct a constant jammer. With frequent jammers, the difference in the performance between fixed rate assignment and rate adaptation is larger, while for a rare jammer it is smaller. This is because with rare jamming, rate adaptation has more time to converge and therefore often succeeds in achieving the highest rate possible; one observes the opposite effect when we have a frequent jammer. The results are plotted in Figs. 6 and 7.

E. Random Jamming in Multihop Topologies

Next, we examine the impact of a random jammer on the end-to-end throughput of a multihop path. We experiment with 15 different routes on our test-bed. We fix static routes

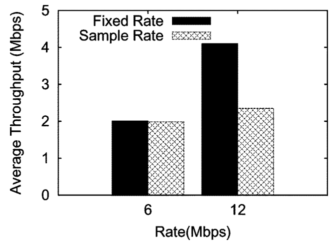


Fig. 8. Rate adaptation presents the same behavior in multihop links. It provides lower throughput at high rates.

of various lengths (from two to four links per route) utilizing the *route* Unix tool in order to modify the routing tables of nodes. We place a jammer such that it affects one or more links. Along each route, links that are not affected by the jammer consistently use a rate adaptation algorithm. *On the links that are subject to jamming, our analysis dictates the decision on whether to use fixed or adaptive rate assignment.* We measure the end-to-end throughput on the route. We show our results for routes on which, in the absence of a jammer, end-to-end throughput of 6 and 12 Mb/s was observed. From Fig. 8, we see that *the throughput trend with rate adaptation on multihop routes in the presence of a random jammer is the same as that on a single-hop link*. In particular, with low data rates, a sufficiently high PDR has to be sustained over the links that constitute the route in order for a fixed-rate approach to perform better than rate adaptation. On the other hand, when routes support high data rates, fixing the rate on the individual links (that are affected by the jammer), as per our analytical framework, provides higher benefits.

F. Choosing the Right Policy in Practice

To summarize our findings, our analysis demonstrates that using a fixed rate may be attractive on lossless links, while it would be better to use rate adaptation on lossy links. However, as discussed, determining when to use one over the other in real time during system operations is difficult. The determination requires the knowledge of $x(F, R_s)$, $y(R_i)$, and estimates of how often the jammer is active/asleep on average. Thus, we choose a simpler practical approach that we call MRC for Markovian rate control. We will describe MRC in detail later (in Section VI), but in a nutshell, MRC induces memory into the system and keeps track of the feasible rates during benign jamming-free periods. As soon as the jammer goes to sleep, legitimate transmissions are invoked at the most recent rate used during the previous sleeping cycle of the jammer. We also perform offline measurements by directly using our analytical formulation (with knowledge of the aforementioned parameters). These measurements serve as benchmarks for evaluating the efficacy of MRC (discussed in Section VII).

V. System Guidelines for Power Control

Next, we examine whether tuning power levels can help cope with the interference injected by a jammer. If we consider a single legitimate data link and a jammer, incrementing the transmission power on the data link should increase the signal-to-interference-plus-noise ratio (SINR) of the received data packets. Thus, one could argue that increasing the transmission power is always beneficial in jamming environments [17]. Note here that increasing the transmission power in environments with lower power jammers can potentially increase the network-wide interference. However, as we will see in the following, ARES includes a CCA tuning mechanism that avoids starvation effects caused from legitimate interference [10].

We vary the transmission powers of both the jammer and legitimate transceiver, as well as the CCA threshold of the latter. Note that the jammer's transmission distribution is not very relevant in this part of our study. Our expectation is that tuning the power of legitimate transceivers will provide benefits while the jammer is active. *In other words, one can expect that the benefits from power control will be similar with any type of jammer.* We define the following:

- $\mathrm{RSSI_{TR}}$: the received signal strength indicator (RSSI) of the signal of the legitimate transmitter at its receiver;
- $\mathrm{RSSI_{RT}}$: the RSSI of the signal in the reverse direction (the receiver is now the transmitter);
- $\mathrm{RSSI_{JT}}$ and $\mathrm{RSSI_{JR}}$: the RSSI values of the jamming signal at the legitimate transmitter and receiver, respectively;
- $\mathrm{RSSI_J}$: the minimum of $\{\mathrm{RSSI_{JT}},\ \mathrm{RSSI_{JR}}\}$;
- P_L and $\mathrm{CCA_L}$: the transmission power and the CCA threshold at legitimate transceivers;
- P_J: the transmission power of the jammer.

Our main observations are the following.

- Mitigating jamming effects by incrementing P_L is viable at low data rates. It is extremely difficult to overcome the jamming interference at high rates *simply with power adaptation*.
- Increasing $\mathrm{CCA_L}$ restores (in most cases) the isolated throughput (the throughput achieved in the absence of jammers).

We present our experiments and the interpretations thereof, in what follows.

A. Increasing P_L to Cope With Jamming Interference

Increasing P_L will increase the SINR, and one might expect that this would reduce the impact of jamming interference on the throughput. In our experiments, we quantify the gains from employing such a "brute-force" approach.

1) Details on the Experimental Process: We perform measurements on 80 different links and with four jammers. We consider different fixed values for P_J (from 1 to 18 dBm). For each of these values, we vary P_L between 1 and 18 dBm and observe the throughput in the presence of the jammer for all possible fixed transmission rates. For each chosen pair of values $\{P_\mathrm{L}, P_\mathrm{J}\}$, we run 60-min repeated experiments and collect a new throughput measurement once every 0.5 s. Both end-nodes of a legitimate link use the same transmission power.

2) Combination of High P_L and Low Data Rate Helps Mitigate the Impact of Low-Power Jammers: We experiment with many different locations of the jammers. Our measurements indicate that when high transmission rates are used, increasing P_L does not help alleviate the impact of jammers. Our results are depicted in Figs. 9 and 10. In these figures, we plot the percentage of the isolated throughput achieved in the presence of jamming for two representative combinations of P_L and P_J and for two different rates. The isolated throughput is the throughput achieved without the presence of jammers. In our experiments on the 80 considered links, *there were no links where incrementing P_L increased the throughput at high data*

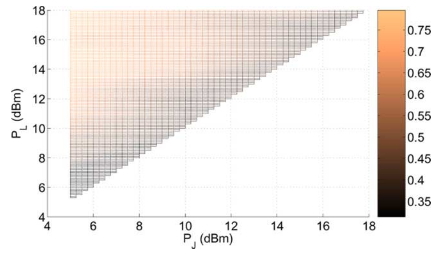


Fig. 9. Percentage of the isolated throughput for various P_{L} and P_{J} combinations and for $R_{\mathrm{f}} = R_{\mathrm{a}} = 6$ Mb/s.

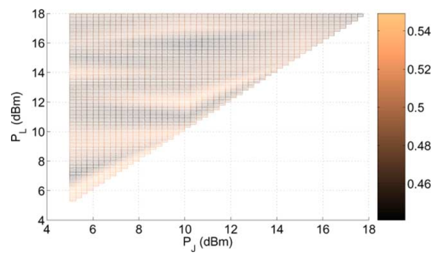


Fig. 10. Percentage of the isolated throughput for various P_{L} and P_{J} combinations and for $R_{\mathrm{f}} = R_{\mathrm{a}} = 54$ Mb/s.

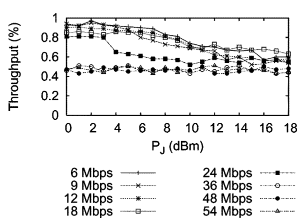


Fig. 11. Percentage of the isolated throughput in the presence of a balanced jammer for various P_{J} and P_{J} values and data rates.

rates, even with very low jamming powers. While there could exist cases where incrementing P_{L} could yield benefits at high rates, this was not observed. In contrast, we observe that with low data rates and when P_{J} is low, data links can overcome jamming to a large extent by increasing P_{L}. Fig. 11 depicts another representative subset of our measurement results where all legitimate nodes use $P_{\mathrm{L}} = 18$ dBm, while P_{J} is varied between 1 and 18 dBm. We observe that the combination of high P_{L} with low data rate helps overcome the impact of jamming, when P_{J} is low. Note also that when P_{J} is high, it is extremely difficult to achieve high average throughput.

The above observations can be explained by taking a careful look at the following two cases.

a) Strong Jammer: Let us consider a jammer such that $\mathrm{RSSI_J} > \mathrm{CCA_L}$. This can result in two effects: 1) The sender will sense that the medium is constantly busy and will defer its packet transmissions for prolonged periods of time. 2) The signals of both the sender and the jammer will arrive at the receiver with RSSI values higher than $\mathrm{CCA_L}$. This will result in a packet collision at the receiver. In both cases, the throughput is degraded. Our measurements show that *it is not possible to mitigate strong jammers simply by increasing* P_{L}.

b) Weak Jammer: Let us suppose that the jammer's signals arrive with low RSSI at legitimate nodes. This may be either due to energy-conservation strategies implemented by the jammer, causing it to use low P_{J} (e.g., 2 dBm) or due to poor channel conditions between a jammer and a legitimate transceiver. At high transmission rates, the SINR required for the successful decoding of a packet is larger than what is required at low rates (shown in Table IV) [10]. Our throughput measurements show that even in the presence of weak jammers, the SINR requirements at high transmission rates are typically not satisfied. However, since the SINR requirements at lower data rates are less stringent, *the combination of high* P_{L} *and low rate provides significant throughput benefits.*

TABLE IV
SINR Levels Required for Successful Packet Decoding in 802.11a/g

Data Rate	6	9	12	18	24	36	48	54
SINR (dB)	6	7.8	9	10.8	17	18.8	24	24.6

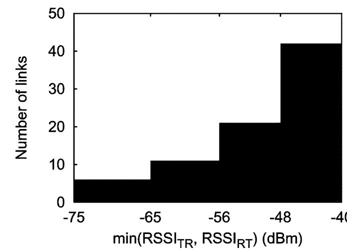


Fig. 12. Histogram with $\mathrm{RSSI_{TR}}$ and $\mathrm{RSSI_{RT}}$ values on legitimate links.

B. Tuning $\mathrm{CCA_L}$ on Single-Hop Settings

Next, we investigate the potential of adjusting $\mathrm{CCA_L}$ in conjunction with P_{L}.

1) Implementation and Experimental Details: For these experiments we exclusively use the *Intel-2915* cards. These cards allow us to tune the CCA threshold. We have modified a prototype version of the AP/client driver in order to periodically collect measurements for $\mathrm{RSSI_{TR}}$, $\mathrm{RSSI_{RT}}$, and $\mathrm{RSSI_J}$. We consider 80 AP–client data links, with traffic flowing from the AP to the client. As before, we divide the 80 data links into 20 sets of four isolated links. Most of these links are strong, with $\min[\mathrm{RSSI_{TR}}, \mathrm{RSSI_{RT}}] > -56$ dBm, as depicted in the histogram in Fig. 12. We use Intel's proprietary rate adaptation algorithm, which has been implemented in the firmware of the *Intel-2915* cards. We measure the achieved data throughput for different values of P_{L} and $\mathrm{CCA_L}$. Both nodes of a data link use the same power and CCA threshold values.

2) Tuning the CCA Threshold Is a Potential Jamming Mitigation Technique: To begin with, we perform throughput measurements with the default $\mathrm{CCA_L}$ value $(-80\,\mathrm{dBm})$ and with various $\mathrm{RSSI_J}$ values. We observe from Fig. 13 that when $\mathrm{RSSI_J} < \mathrm{CCA_L}$, data links achieve high throughput. This is because signals with $\mathrm{RSSI} < \mathrm{CCA_L}$ are ignored by the transceiver's hardware. In particular: 1) such signals do not render the medium busy; and 2) receivers are trying to latch onto signals with $\mathrm{RSSI} > \mathrm{CCA_L}$, while other signals are considered to be background noise. Moreover, even when $\mathrm{RSSI_J}$ is slightly larger than $\mathrm{CCA_L}$, we still observe decent throughput achievements for the cases wherein data links operate at high SINR regimes. This is because the reported RSSI value is an average and the jammer signal could be below the threshold even here, in many cases. These measurements imply that the

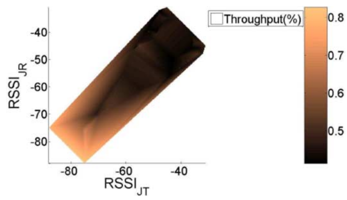


Fig. 13. Percentage of the isolated throughput in the presence of a balanced jammer versus $\mathrm{RSSI_J}$, for $\mathrm{CCA_L} = -80$ dBm.

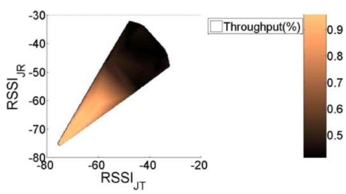


Fig. 14. Percentage of the isolated throughput, for various $\mathrm{RSSI_J}$ values and for $\mathrm{CCA_L} = -50$ dBm.

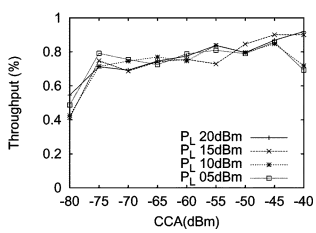


Fig. 15. Percentage of the isolated throughput, for various $\mathrm{CCA_L}$ values and various P_L values. $P_\mathrm{J} = 20$ dBm.

ability to tune $\mathrm{CCA_L}$ can help receive data packets correctly, even while jammers are active.

In order to further explore the potential of such an approach, we vary $\mathrm{CCA_L}$ from -75 to -30 dBm on each of the considered 80 links. Fig. 14 depicts the results for the case where $\mathrm{CCA_L}$ is equal to -50 dBm. We observe that *increasing* $\mathrm{CCA_L}$ *results in significantly higher data throughput, even with quite high* $\mathrm{RSSI_J}$ *values.* More specifically, from Fig. 14 we observe that when $\mathrm{RSSI_J}$ is lower than $\mathrm{CCA_L}$, links can achieve up to 95% of the throughput that is achieved when the medium is jamming free. When $\mathrm{RSSI_J} \approx \mathrm{CCA_L}$, data links still achieve up to 70% of the jamming-free throughput (capture of data packets is still possible to a significant extent). As one might expect, if $\mathrm{RSSI_J} \gg \mathrm{CCA_L}$, there are no performance benefits.

Our observations also hold in some scenarios where, $P_\mathrm{J} > P_\mathrm{L}$. Fig. 15 presents the results from one such scenario. We observe that appropriate CCA settings can allow legitimate nodes to exchange traffic effectively, even when $P_\mathrm{J} \gg P_\mathrm{L}$. This is possible if the link conditions between the jammer and the legitimate transceivers are poor and result in low $\mathrm{RSSI_J}$. Note here that one cannot increase $\mathrm{CCA_L}$ to arbitrarily high values on legitimate nodes. Doing so is likely to compromise connectivity between nodes or degrade the throughput due to failure of capturing packets as seen in Fig. 15 for $P_\mathrm{L} = 5$ dBm and $P_\mathrm{L} = 10$ dBm.

C. *Tuning* $\mathrm{CCA_L}$ *in Multihop Configurations*

We perform experiments with various CCA thresholds along a route. Previous studies have shown that in order to avoid starvation due to asymmetric links, the transmission power and the CCA threshold need to be jointly tuned for all nodes of the same connected (sub)network [10]. In particular, the product $C = P_\mathrm{L} \cdot \mathrm{CCA_L}$ must be the same for all nodes. Given this, we ensure that C is the same for all nodes that are part of a route. In particular, we set P_L to be equal to the maximum possible value of 20 dBm on all nodes of a route. For each run, $\mathrm{CCA_L}$ is therefore set to be the same on all of the nodes on the route. Throughout our experiments with multihop traffic, nodes on one route do not interfere with nodes that are on other routes. In scenarios where nodes belonging to different routes interfere with each other, if all nodes use the same P_L, their $\mathrm{CCA_L}$ values must be the same [10], [30]. However, we did not experiment with such scenarios given that our objective is to isolate the impact of a jammer and not to examine interference between coexisting sessions in a network.

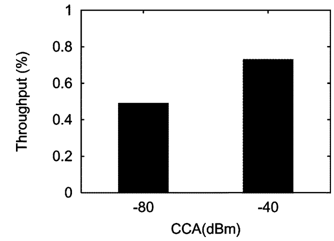


Fig. 16. Careful CCA adaptation significantly improves the end-to-end throughput along a route.

We experiment with the same multihop settings as in Section IV-E. Fig. 16 presents the results observed on one of our routes. We observe that careful CCA tuning can provide significant average end-to-end throughput benefits along a route.[5]

VI. Designing ARES

In this section, we design our system ARES based on the observations from Section V. ARES is composed of two main modules: 1) a *rate module* that chooses between fixed or adaptive-rate assignment; and 2) a *power control module* that facilitates appropriate CCA tuning on legitimate nodes.

A. *Rate Module in ARES*

As discussed in Section IV, our experiments with three popular rate adaptation algorithms show that the convergence time of the algorithms affects the link performance in random-jamming environments. This convergence time is largely implementation-specific. As an example, our experiments with both SampleRate and Onoe show that in many cases it takes more than 10 s for both algorithms to converge to the "best" rate; [31] reports similar observations. The rate module in ARES decides on whether a fixed- or an adaptive-rate approach should be applied.

1) MRC: Markovian Rate Control: MRC is an algorithm-patch that can be implemented on top of any rate control algorithm. MRC is motivated by our analysis in Section IV. However, as discussed earlier, it does not directly apply the analysis since this would require extensive offline measurements (the

[5] We wish to point out that state-of-the-art routing protocols (e.g., [28]) pick routes with links of high quality. If shortest path routing is used instead, links tend to be of poor quality, and CCA tuning may not be easy to implement.

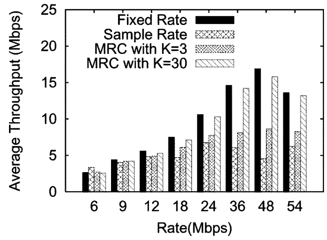


Fig. 17. MRC outperforms current rate adaptation algorithms, especially for high values of K.

collection of which can be time-consuming) and estimates of the jammer active and sleep periods. The key idea that drives MRC is that a rate adaptation algorithm need not examine the performance at all the transmission rates during the sleeping period of the jammer. The algorithm simply needs to remember the previously used transmission rate and use it as soon as the jammer goes to sleep. Simply put, MRC introduces *memory* into the system. The system keeps track of past transmission rates and hops to the stored highest-rate state as soon as the jammer goes to sleep. Since the channel conditions may also change due to the variability in the environment, MRC invokes the rescanning of all rates periodically, once every K consecutive sleeping/jamming cycles. When $K = 1$, we do not expect to have any benefits since the scanning takes place in each cycle.

Note here that the appropriate value of K depends on the environment and the sleep and active periods of the jammer. One could adaptively tune the K value. As an example, an additive increase additive decrease strategy may be used where one would increase the value of K until a degradation is seen. The K value would then be decreased. The implementation of such a strategy, however, is beyond the scope of this paper.

2) Implementation Details of MRC: The implementation: 1) keeps track of the highest transmission rate used over a benign time period (when the jammer is asleep); and 2) applies this rate immediately upon the detection of the next transition from the jammer's active period to the sleeping period.

Fig. 17 presents a set of measurements with MRC, with intermittent SampleRate invocations (once every K cycles) for $K = \{3, 30\}$. We observe that MRC outperforms pure SampleRate in jamming environments, especially with larger values of K. With small K, the rate adaptation algorithm is invoked often, and this reduces the achieved benefits. Furthermore, MRC provides throughput that is close to the maximum achievable on the link (which may be either with fixed or adaptive rate, depending on whether the link is lossy or lossless).

B. Power Control Module in ARES

As discussed in Section V, increasing P_{L} is beneficial at low rates. While at high rates this is not particularly useful, it does not hurt either. Since our goal in this paper is to propose methods for overcoming the effects of jamming (and not legitimate) interference, we impose the use of the maximum P_{L} by all nodes in the presence of jammers. The design of a power control mechanism that in addition takes into account the imposed legitimate interference (due to high P_{L}) is beyond the scope of this paper.

More significantly, our power control module overcomes jamming interference by adaptively tuning $\mathrm{CCA_L}$. The module requires the following inputs on each link.

- The values of $\mathrm{RSSI_{TR}}$, $\mathrm{RSSI_{RT}}$, $\mathrm{RSSI_{JR}}$, and $\mathrm{RSSI_{JT}}$. These values can be easily observed in real time.
- An estimation for the shadow fading variation of the channel, Δ. Due to shadow fading, the above RSSI values can occasionally vary by Δ. The value of Δ is dependent on the environment of deployment. One can perform offline measurements and configure the value of Δ in ARES.

We determine the variations in RSSI measurements via experiments on a large set of links. The measurements indicate that Δ is approximately 5 dB for our test-bed (a less conservative value than what is reported in [32]). The value of $\mathrm{CCA_L}$ has to be at least Δ dB lower than both $\mathrm{RSSI_{TR}}$ and $\mathrm{RSSI_{RT}}$ to guarantee connectivity at all times. Hence, ARES sets

$$\mathrm{CCA_L} = \min(\mathrm{RSSI_{TR}}, \mathrm{RSSI_{RT}}) - \Delta$$

if

$$\max(\mathrm{RSSI_{JT}}, \mathrm{RSSI_{JR}}) \leq \min(\mathrm{RSSI_{TR}}, \mathrm{RSSI_{RT}}) - \Delta.$$

Otherwise, $\mathrm{CCA_L}$ is not changed.[6] This ensures that legitimate nodes are always connected, while the jammer's signal is ignored to the extent possible. Our experiments indicate that especially if

$$\max(\mathrm{RSSI_{JT}}, \mathrm{RSSI_{JR}}) \leq \min(\mathrm{RSSI_{TR}}, \mathrm{RSSI_{RT}}) - 2\Delta$$

the data link can operate as if it is jamming-free.

In order to avoid starvation effects, the tuning of the CCA threshold should be performed only when nodes that participate in power control belong to the same network [30]. Unless collocated networks cooperate in jointly tuning their CCA (as per our scheme), our power control module will not be used. Note that when jamming attacks become more prevalent, cooperation between coexisting networks may be essential in order to fight the attackers. Hence, in such cases, collocated networks can have an agreement to jointly increase the CCA thresholds when there is a jammer.

1) Implementation Details: Our power control algorithm can be applied in a *centralized* manner by having all legitimate nodes report the required RSSI values to a central server. The central server then applies the same $\mathrm{CCA_L}$ value to all nodes (of the same connected network). The chosen $\mathrm{CCA_L}$ is the highest possible CCA threshold that guarantees connectivity between legitimate nodes. This reporting requires trivial modifications on the wireless drivers. We have implemented a centralized functionality when our network is configured as a multihop wireless mesh.

In a *distributed* setting, our algorithm is applicable as long as legitimate nodes are able to exchange RSSI information. Each node can then independently determine the $\mathrm{CCA_L}$ value. To demonstrate its viability, we implement and test a distributed version of the power control module in an 802.11a/g WLAN configuration. In particular, we modify the Intel prototype AP driver by adding an extra field in the "Beacon" template. This new field contains a matrix of RSSI values of neighboring jammers and legitimate nodes. We enable the decoding of received beacons in the AP driver (they do not read these by default). Assuming that a jammer imposes almost the same amount of interference on all devices (AP and clients) within a cell, the AP

[6]We choose not to tune $\mathrm{CCA_L}$ unless we are certain that it can help alleviate jamming interference.

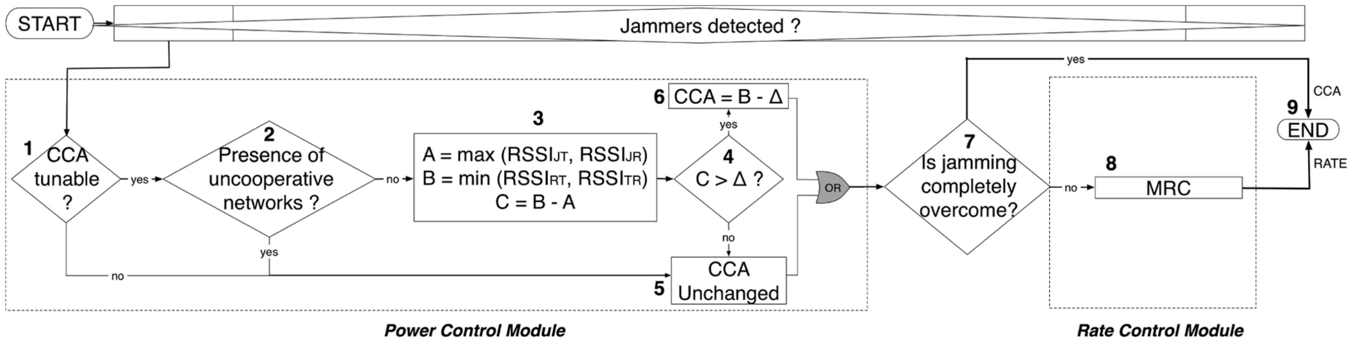


Fig. 18. ARES, our Anti-jamming REinforcement System.

of the cell determines the final $\mathrm{CCA_L}$ after a series of iterations in a manner very similar to the approaches in [10] and [30].

C. Combining the Modules to Form ARES

We combine our rate and power control modules to construct ARES as shown in Fig. 18. The goal of ARES is to apply the individual modules as appropriate once the jammers are detected. For the latter, ARES relies on already existing jamming detection schemes and inherits their accuracy. For example, the mechanism that was proposed in [14] can be used; this functionality performs a consistency check between the instantaneous PDR and RSSI values. If the PDR is extremely low while the RSSI is much higher than the default $\mathrm{CCA_L}$, the node is considered to be jammed. We want to reiterate that it is beyond the scope of our work to design a new, even more accurate, detection scheme.

ARES applies the power control module first since, with this module, the impact of the jammer(s) could be completely overcome. If the receiver is able to capture and decode all packets in spite of the jammer's transmissions, no further actions are required. Note that even if $\mathrm{CCA_L} > \mathrm{RSSI_J}$, the jammer can still affect the link performance. This is because with CCA tuning the jamming signal's power is added to the noise power. Hence, even though the throughput may increase, the link may not achieve the "jamming-free performance" while the jammer is active. If the jammer still has an effect on the network performance after tuning $\mathrm{CCA_L}$ (or if CCA tuning is infeasible due to the presence of collocated uncooperative networks), ARES enables the rate module. Note that the two modules can operate independently, and the system can bypass any of them in case the hardware/software does not support the specific functionality.

VII. Evaluating Our System

We first evaluate ARES by examining its performance in three different networks: a MIMO-based WLAN, an 802.11 mesh network in the presence of mobile jammers, and an 802.11a WLAN setting where uplink TCP traffic is considered.

ARES boosts the throughput of our MIMO WLAN under jamming by as much as 100%. Our objective here is twofold. First, we seek to observe and understand the behavior of MIMO networks in the presence of jamming. Second, we wish to measure the effectiveness of ARES in such settings. Toward this, we deploy a set of seven nodes equipped with *Ralink RT2860* miniPCI cards.

A. Experimental Setup

We examine the case for a WLAN setting since the *RT2860* driver does not currently support the *ad hoc* mode of operations. MIMO links with space–time block codes (STBCs) are expected to provide robustness to signal variations, thereby reducing the average SINR that is required for achieving a desired bit error rate, as compared to a corresponding SISO link. For our experiments, we consider two APs, with two and three clients each, and two jammers. Fully saturated downlink UDP traffic flows from each AP to its clients.

B. Applying ARES on a MIMO-Based WLAN

We first run experiments without enabling ARES. Interestingly, we observe that in spite of the fact that STBC is used, 802.11n links present the same vulnerabilities as 802.11a or g links. In other words, MIMO does not offer significant benefits by itself in the presence of a jammer. This is due to the fact that 802.11n is still employing CSMA/CA, and as a result, the jamming signals can render the medium busy for a MIMO node as well. Moreover, for STBCs to work effectively and provide a reduction in the SINR for a desired bit error rate (BER), the signals received on the two antenna elements will have to experience independent multipath fading effects. In other words, a line of sight or dominant path must be absent. However, in our indoor test-bed, given the proximity of the communicating transceiver pair, this may not be the case. Thus, little diversity is achieved [33] and does not suffice in coping with the jamming effects.

Next, we apply ARES and observe the behavior. The logical set of steps that ARES follows (in Fig. 18) is $1 \rightarrow 5 \rightarrow 7 \rightarrow 8 \rightarrow 9$. Since the CCA threshold is not tunable with the *RT2860* cards, ARES derives decisions with regard to rate control only. Fig. 19 depicts the results. We observe that the configuration with ARES outperforms the rate adaptation scheme that is implemented on the *RT2860* cards in the presence of the jammer by as much as 100%. Note that higher gains would be possible if ARES were able to invoke the power control module.

In Fig. 19, we also compare the throughput with MRC to the suggested settings with our analysis (these settings allow us to obtain benchmark measurements possible with global information). The parameters input to the analysis are the following.

1) The jammer is balanced with a jamming distribution $U[1, 5]$ and a sleep distribution $U[1, 6]$.
2) We examine four R_a values: 13.5, 27, 40.5, and 54 Mb/s.
3) $F = 0$ Mb/s.
4) We input estimates of the $y(R_i)$ values that are obtained via comprehensive offline measurements.

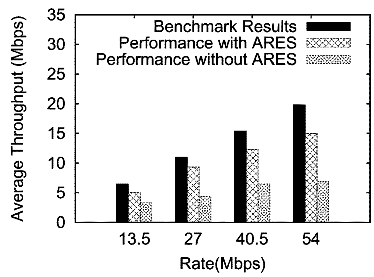


Fig. 19. ARES provides significant throughput benefits in a MIMO network in the presence of jammers.

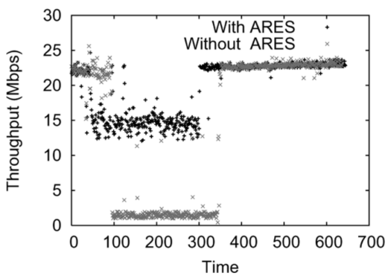


Fig. 20. ARES provides significant throughput improvement in mobile-jamming scenarios.

5) The offline measured $\mathrm{PDR_f}$.

We observe that the performance with MRC is quite close to our benchmark measurements. These results show that in spite of having no information with regards to the jammer distribution or the convergence times of the rate adaptation algorithms, MRC is able to significantly help in the presence of a random jammer.

1) ARES Increases the Link Throughput by up to 150% in an 802.11a Mesh Deployment With Mobile Jammers: Next, we apply ARES in an 802.11a mesh network with mobile jammers and UDP traffic. We consider a frequent jammer (jamming distribution $U[1,20]$ and sleeping distribution $U[0,1]$). The jammer moves toward the vicinity of the legitimate nodes, remains there for k s, and subsequently moves away. For the mobile jammer, we used a laptop, equipped with one of our Intel cards, and carried it around. The power control module is implemented in a centralized manner. ARES increases $\mathrm{CCA_L}$ in order to overcome the effects of jamming interference to the extent possible. In this case, due to the aggressiveness of the considered jammer (prolonged jamming duration), the rate adaptation module does not provide any benefits (since rate control helps only when the jammer is sleeping). In this scenario, ARES follows the steps $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9$. Fig. 20 depicts throughput-time traces, with and without ARES, for an arbitrarily chosen link and $k \approx 200$. The use of ARES tremendously increases the link throughput during the jamming period (by as much as 150%). We have observed the same behavior with a distributed implementation of the power control module in an 802.11a WLAN setting.

2) ARES Improves the Total AP Throughput by up to 130% With TCP Traffic: Next, we apply ARES on an 802.11a WLAN. For this experiment, we use nodes equipped with the *Intel-2915* cards. We consider a setting with one AP and two clients, where clients can sense each others' transmissions. We place a balanced jammer (jamming distribution $U[1,5]$ and sleeping $U[1,8]$) such that all three legitimate nodes can sense its presence. We enable fully saturated uplink TCP traffic from all clients to the AP (using *iperf*), and we measure the total throughput at the AP, once every 0.5 s. In this scenario, ARES follows the logical steps $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9$. From Fig. 21, we observe that *the total AP throughput is improved by up to 130% during the periods that the jammer is active*. The benefits are less apparent when the jammer is sleeping because TCP's own congestion control algorithm is unable to fully exploit the advantages offered by the fixed rate strategy.

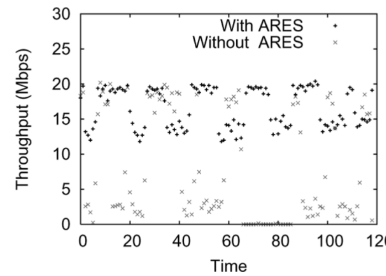


Fig. 21. ARES improves the client–AP link throughput by 130% with TCP traffic scenarios.

3) Applying MRC on an AP Improves the Throughput of Neighbor APs by as Much as 23%: With MRC, a jammed node utilizes the lowest rate (when the jammer is active) and highest rate (when the jammer is sleeping) that provide the maximum long-term throughput. With this, the jammed node avoids examining the intermediate rates and, as we showed above, this increases the link throughput. We now examine how this rate adaptation strategy affects the performance of neighbor legitimate nodes. We perform experiments on a topology consisting of four APs and eight clients, with two clients associated with each AP, all set to 802.11a mode. A balanced jammer with a jamming distribution $U[1,5]$ and a sleep distribution $U[1,6]$ is placed such that it affects *only* one of the APs. Only the affected AP is running MRC; the rest of the APs use SampleRate. We activate different numbers of APs at a time, and we enable fully saturated downlink traffic from the APs to their clients. Fig. 22 depicts the average total AP throughput. Interestingly, we observe that *the use of MRC on jammed links improves the performance of neighbor APs that are not even affected by the jammer*. This is because the jammed AP does not send any packets using intermediate bit rates (such as with the default operation of rate adaptation algorithms). Since MRC avoids the transmission of packets at lower (than the highest sustained) bit rates, the jammed AP does not occupy the medium for as prolonged periods as with the default rate control techniques; the transmission of packets at the high rate (while the jammer is asleep) takes less time. Hence, this provides more opportunities for neighbor APs to access the medium, thereby increasing the AP throughput. Specifically, we observe that the throughput of one neighbor AP is improved by 23% (when the topology consists of only two APs, one of which is jammed). As we further increase the number of neighbor APs, the benefits due to MRC are less pronounced due to increased contention (Fig. 22).

4) ARES Converges Relatively Quickly: Finally, we perform experiments to assess how quickly the distributed form of ARES converges to a rate and power control setting. In a nutshell, our implementation has demonstrated that the network-wide convergence time of ARES is relatively small. With MRC, the rate control module can very rapidly make a decision with regards to

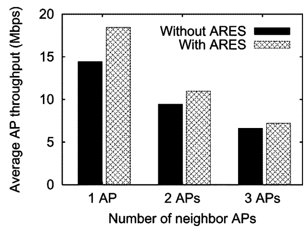


Fig. 22. MRC improves the throughput of neighbor legitimate devices, as compared to SampleRate.

TABLE V
AVERAGE CONVERGENCE TIMES (IN SECONDS) FOR DIFFERENT P_{J} VALUES

P_J (dB)	Convergence time (sec)
1	1.8
2	2.4
3	2.8
4	3.5

the rate setting; as soon as the jammer is detected, MRC applies the appropriate stored lowest and highest rates.

With regard to the convergence of the power control module, recall that our implementation involves the dissemination of the computed CCA value through the periodic transmission of beacon frames (one beacon frame per 100 ms is transmitted with our ipw2200 driver) [30]. As one might expect, the jammer's signal may collide with beacon frames, and this makes it more difficult for the power control module to converge. Note also that as reported in [30] and [34], beacon transmissions are not always timely, especially in conditions of high load and poor-quality links (such as in jamming scenarios). We measure the network-wide convergence time, i.e., the time elapsed from the moment that we activate the jammer until all legitimate devices have adjusted their CCA threshold as per our power control scheme. First, we perform measurements on a multihop mesh topology consisting of five APs and 10 clients (two clients per AP). In order to have an idea about whether the observed convergence time is significant, we also perform experiments without jammers, wherein we manually invoke the power control module through a user-level socket interface on one of the APs. We observe that the convergence time for the specific setting is approximately 1.2 s. Then, we activate a continuously transmitting *deceptive* jammer in a close proximity to two neighbor APs (MRC is disabled; the jammer affects only the two APs). Table V contains various average convergence times for the specific setting and for different P_{J} values.

We observe that although the convergence time increases due to jamming, it still remains short. Furthermore, we perform extensive experiments with eight APs, 19 clients, and four balanced jammers with $P_{\mathrm{J}} = 3$ dBm, all uniformly deployed. We observe that in its distributed form, the power control module converges in approximately 16 s in our network-wide experiments. Although one may expect different (lower or higher) convergence times with different hardware/software and/or mobile jammers, these results show that in a static topology the power control module converges relatively quickly in practical settings. Note here that, the convergence time of the power control module appears to be larger as compared to that of the conventional rate control algorithms. However, there is a fundamental difference between the two functionalities that makes the slower convergence time of the distributed power control module less harmful. In particular, once ARES has identified the *right* CCA settings, it can quickly apply them in every consequent jamming cycle. On the contrary, the rate control algorithms (e.g., SampleRate) will need to start their search for the best rate in every jamming/sleeping cycle.

VIII. SCOPE OF ARES

From our evaluations, it is evident that ARES can provide performance benefits in the presence of jamming, even with other wireless technologies, and both in static and dynamically changing environments. In this section, we discuss some design choices and the applicability requirements of ARES.

A. ARES Requires a Driver/Firmware Update, but Does Not Necessitate Changes in 802.11.

The two modules that constitute ARES are relatively easy to implement in the driver/firmware of commodity wireless cards and do not require any hardware changes. The only software modification needed in the firmware involves the CCA tuning functionality. Specifically, it should be possible to change the CCA threshold as per the commands sent through a driver-firmware socket interface. To facilitate a distributed WLAN implementation of ARES, the AP driver needs to be modified to read the new Beacon template from the Beacons received from neighbor cochannel APs. Finally, clients need to apply the power and CCA settings determined by their affiliated AP. Deployment of ARES will require a driver/firmware upgrade of wireless NICs; this involves the cooperation of both vendors and users of the existing cards. Note here that devices that have not performed the ARES firmware update can still operate with those with the update. However, the performance gains of our system will diminish. Finally, note that our system does not require any changes to the 802.11 standard. In particular, the standard: 1) does not specify values for the CCA, transmission power, and rate; and 2) leaves space in the Beacon's template for communicating additional information required. Thus, ARES is compliant with 802.11 operations.

B. On the Effectiveness of MRC

Our analysis provided in Section IV is an accurate tool that decides between the use of a fixed rate or a rate adaptation strategy. However, applying the analysis in a real system is quite challenging for various reasons. In particular, as discussed earlier, the analysis requires a set of inputs that may not be readily available. If the analysis were to be applied in real time, ARES would need to observe these values on the fly and invoke the rate module whenever significant nontemporal changes are observed. It is also difficult to derive the jammer's distribution accurately and quickly. Such requirements make the application of the analysis somewhat infeasible in real-time systems. Furthermore, the analysis can account for the presence of one jammer only. In scenarios with multiple jammers, it cannot decide between fixed or adaptive rate.

In contrast, our more practical scheme MRC does not need any inputs. It can operate efficiently even with multiple jammers. Note that MRC in its current form takes into account the time[7] that has elapsed since the last time that rate control was invoked. The policy is to invoke the rate adaptation strategy

[7]In its current form, this time is in terms of the number of jamming cycles; this can be easily modified to use more generic time units.

after periodic intervals. The optimal rate at which rate adaptation should be invoked depends on the temporal variability of the channel. In particular, to perform this optimally, ARES would need to measure (or estimate) the coherence time τ of the channel (time for which the channel remains unchanged [35]) and invoke the rate control algorithm every τ s. While this is not possible with current 802.11 hardware, it may be possible in the future [35]. Alternatively, ARES could employ a learning strategy as discussed in Section VI.

C. ARES With Reactive and Constant Jammers

For the most part in this paper, we considered various types of random jammers. With constant jammers, rate adaptation is not expected to provide benefits since the continuous jamming interference does not allow the use of high rates. Nevertheless, rate control (even as a *standalone* module) is expected to provide benefits in the presence of reactive jamming. In particular, let us consider a link consisting of legitimate nodes A and B. The reactive jammer J needs to sense the ongoing transmission and quickly transmit its jamming signal. If we denote by $t_{\rm flight}$ the flight time of the legitimate packet, and with $t_{\rm sense}$ the time needed for J to sense this packet, then the probability of successful packet corruption[8] can be calculated as $P_{\rm jam} = P(t_{\rm sense} < t_{\rm flight})$. Assuming that $t_{\rm sensing}$ is uniformly distributed at the interval $[0, {\rm DIFS}]$,[9] we get

$$P_{\rm jam} = \int_0^{t_{\rm flight}} \frac{1}{\rm DIFS} dt = \frac{t_{\rm flight}}{\rm DIFS} = \frac{\#{\rm bytes/packet}}{{\rm rate} \cdot {\rm DIFS}}. \quad (2)$$

From (2), it is clear that through the use of high bit rates and/or reduced packet sizes, the probability of successful reactive jamming can be decreased. However, there is a tradeoff between successful reception and decreased jamming probability that needs to be examined more carefully. Finally, the power control module of ARES can be useful in the presence of both constant (as shown in Section VII) and reactive jamming.

D. Rate Control Algorithms Dealing With Hidden Terminals Cannot Mitigate Jamming Attacks

There are a few rate control algorithms that try to deal with hidden terminal nodes in WLANs. Their key feature is distinguishing between packet errors due to: 1) channel effects and 2) collisions. As an example, Kim *et al.* [36] propose collision-aware rate adaptation (CARA), which makes use of the RTS/CTS exchange to differentiate between a channel error and a collision. In brief, a packet loss after a successful RTS/CTS exchange will be most probably due to channel errors. However, even if the operations of a jammer cause effects that resemble those due to hidden terminals,[10] CARA is not particular useful. The reason is that the jammer does not follow the rules specified by the 802.11 standard. As a result, even if it listens to the RTS/CTS exchange (which can be successful even under jamming due to their short durations), it will not back off, and thus there will still be a packet collision. CARA will assume that this is due to channel errors, and a less aggressive modulation scheme will be used, reducing the transmission rate. Similar techniques are employed by RRAA [21], and even this scheme cannot efficiently cope with jamming attacks. In addition, RRAA is implemented using specialized programmable hardware platforms with capabilities that most of the commodity APs do not have.

[8]We assume an optimal reactive jammer, i.e., one that is able to jam at the exact time instance when it senses a legitimate packet (best-case scenario for the adversary). In reality, this will not be the case.

[9]This is a reasonable assumption to make since the protocol allows for a DIFS period in order to sense any transmission.

[10]Both ignore ongoing transmissions—the jammer because it simply wants to, while the hidden terminal because it cannot listen to them.

E. Limitations of ARES

ARES's design principles are based on the operations of the 802.11 protocol. Thus, its applicability is limited to this "class" of networks. Extending our system to broadly support any wireless network is not straightforward. However, our goal in this work is the design of a jamming countermeasure for 802.11. In addition, we acknowledge that ARES utilizes functionalities that are currently unavailable in commercial NICs. In particular, CCA tuning (and, in general, the carrier sensing functionality of the protocol) is implemented in the firmware of the cards, which is generally closed-source. Therefore, its adoption is limited to systems that provide access to these operations. However, we expect that given its benefits (and the emergence of software-defined radios), this functionality will be made available as a tunable feature.

IX. Conclusion

We design, implement, and evaluate ARES, an anti-jamming system for 802.11 networks. ARES has been built based on observations from extensive measurements on an indoor test-bed in the presence of random jammers and is primarily composed of two modules. The *power control module* tunes the CCA thresholds in order to allow the transmission and capture of legitimate packets in the presence of the jammer's signals to the extent possible. The *rate control module* decides between fixed- or adaptive-rate assignment. We demonstrate the effectiveness of ARES in three different deployments: 1) an 802.11n-based MIMO WLAN; 2) an 802.11a network infested with mobile jammers; and 3) an 802.11a WLAN with uplink TCP traffic. ARES can be used in conjunction with other jamming mitigation techniques (such as frequency hopping or directional antennas). Overall, the application of ARES leads to significant performance benefits in jamming environments.

Acknowledgment

The authors thank Ralink Corporation, Cupertino, CA, for providing the source of the *RT2860* AP driver, and Dr. K. Papagiannaki from Intel Research, Pittsburgh, PA, for providing the prototype version of the *ipw2200* driver.

References

[1] "SESP jammers," SESP [Online]. Available: http://www.sesp.com/
[2] "Wireless noise hampers DefCon; Impact of non-Wi-Fi interference surprises observers at hacker conference," *BNET.com* 2005 [Online]. Available: http://findarticles.com/p/articles/mi_m0EIN/is_2005_August_2/ai_n14841565
[3] "Techworld news," [Online]. Available: http://www.techworld.com/mobility/news/index.cfm?newsid=10941
[4] "RF jamming attack," ZOHO Corporation, Pleasanton, CA, 2010 [Online]. Available: http://manageengine.adventnet.com/products/wifi-manager/rfjamming-attack.html
[5] 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, pp. 2526–2530.

[6] W. Hu, T. Wood, W. Trappe, and Y. Zhang, "Channel surfing and spatial retreats: Defenses against wireless denial of service," in *Proc. ACM WiSe*, 2004, pp. 80–89.
[7] "ISM wide-band jammers," [Online]. Available: http://69.6.206.229/e-commerce-solutions-catalog1.0.4.html
[8] D. Caro, "[ISN] Users fear wireless networks for control," 2007 [Online]. Available: http://lists.jammed.com/ISN/2007/05/0122.html
[9] K. Pelechrinis, C. Koufogiannakis, and S. V. Krishnamurthy, "Gaming the jammer: Is frequency hopping effective?," in *Proc. WiOpt*, Jun. 2009, pp. 187–196.
[10] V. Mhatre, K. Papagiannaki, and F. Baccelli, "Interference mitigation through power control in high density 802.11 WLANs," in *Proc. IEEE INFOCOM*, 2007, pp. 535–543.
[11] J. Bicket, "Bit-rate selection in wireless networks," M.S. thesis, Dept. Elect. Eng. Comput. Sci., MIT, Cambridge, MA, 2005.
[12] "Onoe rate control," [Online]. Available: http://madwifi.org/browser/trunk/ath_rate/onoe
[13] S. Pal, S. R. Kundu, K. Basu, and S. K. Das, "IEEE 802.11 rate control algorithms: Experimentation and performance evaluation in infrastructure mode," in *Proc. PAM*, 2006.
[14] 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, pp. 46–57.
[15] R. Gummadi, D. Wetheral, B. Greenstein, and S. Seshan, "Understanding and mitigating the impact of RF interference on 802.11 networks," in *Proc. ACM SIGCOMM*, 2007, pp. 385–396.
[16] W. Hu, K. Ma, W. Trappe, and Y. Zhang, "Jamming sensor networks: Attacks and defense strategies," *IEEE Netw.*, vol. 20, no. 3, pp. 41–47, May–Jun. 2006.
[17] G. Lin and G. Noubir, "On link layer denial of service in data wireless LANs," *Wireless Commun. Mobile Comput.*, vol. 5, no. 3, pp. 273–284, May 2003.
[18] G. Noubir and G. Lin, "Low-power DoS attacks in data wireless LANs and countermeasures," presented at the ACM MobiHoc, 2003, Poster.
[19] G. Noubir, "On connectivity in Ad Hoc network under Jamming using directional antennas and mobility," *Wired/Wireless Internet Commun.*, vol. 2957/2004, pp. 186–200, 2004.
[20] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of service attacks in wireless networks: The case of jammers," *IEEE Commun. Surveys Tutorials*, 2011, to be published.
[21] S. Wong, H. Yang, S. Lu, and V. Bharghavan, "Robust rate adaptation in 802.11 wireless networks," in *Proc. ACM MobiCom*, 2006, pp. 146–157.
[22] V. Shah and S. V. Krisnamurthy, "Handling asymmetry in power heterogeneous ad hoc networks: A cross layer approach," in *Proc. IEEE ICDCS*, 2005, pp. 749–759.
[23] H. Zhai and Y. Fang, "Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–12.
[24] I. Broustis, J. Eriksson, S. V. Krishnamurthy, and M. Faloutsos, "A blueprint for a manageable and affordable wireless testbed: Design, pitfalls and lessons learned," in *Proc. IEEE TridentCom*, 2007, pp. 1–6.
[25] "Wireless networking research testbed," Univ. California, Riverside, CA, 2009 [Online]. Available: http://networks.cs.ucr.edu/testbed
[26] "The MadWiFi project," [Online]. Available: http://madwifi-project.org/
[27] "RT2860 wireless driver," [Online]. Available: http://www.ralinktech.com/ralink/Home/Support/Linux.html
[28] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high throughput path metric for multi-hop wireless routing," in *Proc. ACM MobiCom*, 2003, pp. 134–146.
[29] J. C. Chen and J. M. Gilbert, "Measured performance of 5-GHz 802.11 a wireless LAN systems," Atheros Communications, San Jose, CA, White Paper, Aug. 2001.
[30] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. Mhatre, "MDG: Measurement-driven guidelines for 802.11 WLAN design," in *Proc. ACM MobiCom*, 2007, pp. 254–265.
[31] "SampleRate bug," [Online]. Available: http://madwifi.org/ticket/989
[32] S. Zvanovec, P. Pechac, and M. Klepal, "Wireless LAN networks design: Site survey or propagation models?," *Radioengineering*, vol. 12, pp. 42–49, Dec. 2003.
[33] H. Jafarkhani, *Space-Time Coding: Theory and Practice*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[34] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating access point selection in IEEE 802.11 wireless networks," in *Proc. ACM IMC*, 2005, p. 26.
[35] J. Camp and E. W. Knightly, "Modulation rate adaptation in urban and vehicular environments: Cross-layer implementation and experimental evaluation," in *Proc. ACM MobiCom*, 2008, pp. 315–326.
[36] J. Kim, S. Kim, S. Choi, and D. Qiao, "CARA: Collision-aware rate adaptation for IEEE 802.11 WLANs," in *Proc. IEEE INFOCOM*, 2006, pp. 1–11.

Konstantinos Pelechrinis (S'07–M'10) received the Diploma in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2006, and the M.Sc. and Ph.D. degrees in computer science from the University of California, Riverside, in 2008 and 2010, respectively.

He has been an Assistant Professor with the School of Information Sciences faculty, University of Pittsburgh, Pittsburgh, PA, since Fall 2010. He has also held research positions at Los Alamos National Laboratory, Los Alamos, NM; Thomson Research Labs Paris, Paris, France; and Microsoft Research Cambridge, Cambridge, U.K. He was a Visiting Researcher with the University of Thessaly, Volos, Greece, during Fall 2008. His research interests include wireless networking, especially security-related issues that span the full protocol stack. He is involved in protocol design, real-world experimentation, and performance analysis. He is also interested in mathematical foundations of communication networks.

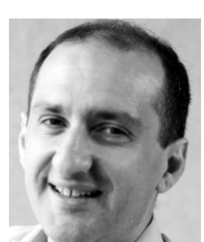
Ioannis Broustis (M'08) received the Diploma in electronics and computer engineering from the Technical University of Crete, Chania, Greece, in 2003, and the M.Sc. and Ph.D. degrees in computer science and engineering from the University of California, Riverside, in 2005 and 2007, respectively.

He has been with Alcatel-Lucent, Murray Hill, NJ, since January 2010. During 2009, he was a Researcher with the University of California, Riverside. During 2008, he was a Researcher with the Center for Research and Technology Hellas, Thessaloniki, Greece, as well as an Adjunct Assistant Professor with the University of Thessaly, Volos, Greece. During 2006, he was an intern with Intel Research, Cambridge, U.K. During 2005, he was an intern with Nokia R&D, Boston, MA. His research interests include cross-layer network protocol design, development, and test-bed experimentation for wireless networks.

Srikanth V. Krishnamurthy (S'94–M'00–SM'07) received the Ph.D. degree in electrical and computer engineering from the University of California, San Diego, in 1997.

From 1998 to 2000, he was a Research Staff Scientist with the Information Sciences Laboratory, HRL Laboratories, LLC, Malibu, CA. Currently, he is a Professor of computer science with the University of California, Riverside. He was the Editor-in-Chief for *Mobile Computing and Communications Review* from 2007 to 2009. His research interests are primarily in wireless networks and security.

Dr. Krishnamurthy is the recipient of the National Science Foundation (NSF) CAREER Award from ANI in 2003.



Christos Gkantsidis (M'05) received the Bachelor's degree from the University of Patras, Patra, Greece, in 1999, and the Ph.D. degree from the Georgia Institute of Technology, Atlanta, in 2006, both in computer science.

He is a Researcher with the Systems and Networking Group, Microsoft Research Cambridge, Cambridge, U.K. He is interested in content distribution networks, peer-to-peer technologies, analysis and modeling of complex communication networks, and wireless mesh networking.

Dr. Gkantsidis is a Member of the Association for Computing Machinery (ACM).