A Software Framework for Alleviating the Effects of MAC-aware Jamming Attacks in Wireless Access Networks

**Ioannis Broustis · Konstantinos Pelechrinis ·
Dimitris Syrivelis · Srikanth V. Krishnamurthy ·
Leandros Tassiulas ·**

Abstract The IEEE 802.11 protocol inherently provides the same long-term throughput to all the clients associated with a given access point (AP). In this paper, we first identify a clever, low-power jamming attack that can take advantage of this behavioral trait: the placement of a low-power jammer in a way that it affects a single legitimate client can cause starvation to all the other clients. In other words, the total throughput provided by the corresponding AP is drastically degraded. To fight against this attack, we design FIJI, a cross-layer anti-jamming system that detects such intelligent jammers and mitigates their impact on network performance. FIJI looks for anomalies in the AP load distribution to efficiently perform jammer detection. It then makes decisions with regards to *optimally* shaping the traffic such that: (a) the clients that are not explicitly jammed are shielded from experiencing starvation and, (b) the jammed clients receive the maximum possible throughput under the given conditions. We implement FIJI in real hardware; we evaluate its efficacy through experiments on two wireless testbeds, under different traffic scenarios, network densities and jammer locations. We perform experiments both indoors and outdoors, and we consider both WLAN and mesh deployments. Our measurements suggest that FIJI detects such jammers in real-time and alleviates their impact by allocating the available bandwidth in a fair and efficient way.

This work was done partially with support from the US Army Research Office under the Multi-University Research Initiative (MURI) grants W911NF-07-1-0318 and the NSF NeTS:WN / Cyber trust grant 0721941.

I. Broustis
University of California, Riverside
(currently at Alcatel-Lucent, USA)
E-mail: broustis@cs.ucr.edu

K. Pelechrinis
University of Pittburgh
E-mail: kpele@pitt.edu

S. V. Krishnamurthy
University of California, Riverside
E-mail: krish@cs.ucr.edu

D. Syrivelis, L. Tassiulas
University of Thessaly
E-mail: {jsyr, leandros}@uth.gr

Keywords IEEE 802.11 Wireless Medium Access Control, Fairness, Jamming, Implementation, Testbed, Measurement.

1 Introduction

The proliferation of IEEE 802.11-based networks makes them an attractive target for malicious attackers with jamming devices [1,2]. A jammer typically emits electromagnetic energy thereby causing: *(a)* prolonged packet collisions at collocated devices, and *(b)* packet transmission deferrals due to legitimate nodes detecting continuous medium activity. Hence, jamming attacks can lead to significant throughput degradation, especially when they intelligently exploit the properties of the MAC protocol in use.

In this paper, we first identify a clever jamming attack where the jammer can not only hurt its intended victim, but cause starvation to other clients that are associated with the same AP as the victim. We call this attack the *Implicit-Jamming* attack. We design and implement FIJI, a cross-layer anti-jamming system to effectively detect such jammers and mitigate the impact of their attack.

The implicit-jamming attack: An inherent characteristic of the IEEE 802.11 MAC protocol is that under saturated traffic demands, an AP (access point) will provide the *same* long-term throughput to all of its affiliated clients [3]. If a client cannot receive high throughput from its AP for any reason (e.g. long-distance AP$\rightarrow$client link or high levels of interference at the client side), the AP will spend a large amount of time serving this client at a low transmission bit-rate; this rate is determined by the rate adaptation algorithm in use. This will compel the AP to serve each of its other "healthier" clients (to which it can support higher transmission rates) for smaller periods. In other words, the AP does not distinguish between clients with low-SINR links and clients with high-SINR links; the long times taken to serve the former class of clients hurts the time available to serve the latter class of clients. This behavior is referred to as *the performance anomaly* of 802.11 [4] and is caused by the inherent design principles of the IEEE 802.11 MAC protocol (described in more detail in section 2).

The implicit jammer exploits this anomaly. To illustrate, consider the WLAN scenario depicted in Fig. 2 (left). In this scenario: *(a)* all 5 clients have high-SINR links with their AP in benign conditions, and *(b)* a low power jammer is placed next to a particular client (client C) such that it does not *directly* affect any other client of the AP. The jammer causes high levels of interference at client C and thus, most of the packets sent by the AP to C are not successfully received. This in turn causes the AP to reduce the transmission rate used to serve C (an inherent property of rate adaptation). As a result, the AP spends more time attempting to serve C, and this reduces the fraction of time that it provides to its other clients. Thus, the throughput of all the clients drops significantly due to the jamming of only client C. In other words, jamming a small subset of clients (even only a single client) implicitly affects all the clients that are affiliated with the same AP. Furthermore, depending on the traffic pattern and the topology, the implicit jamming attack may affect a much wider part of the network than a limited WLAN setting. As an example, consider the mesh deployment in Fig. 2 (right), where all devices are set to the same channel. Let us assume that clients S_1-S_4 transmit fully saturated UDP traffic to clients D_1-D_4 respectively, via a mesh backhaul network consisting of a set of wireless routers [5]. Similarly as above, a low-power jammer is placed next to client D_3 such that only the latter is explicitly affected. Due to the fair nature of 802.11, AP_3 needs to spend a lot more time serving client D_3 and

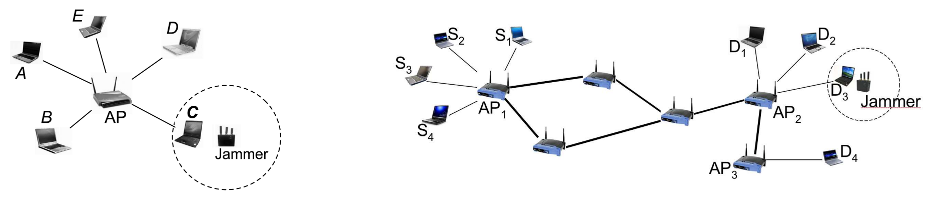

Fig. 1 **Implicit Jamming:** The jammer takes advantage of the 802.11 performance anomaly. Using very low transmission power, it simply attacks one client. The attack can affect a WLAN topology (left) as well as an extended mesh deployment (right).

hence, the fraction of time allocated for transmissions to clients D_1-D_3 as well as to AP_3 is reduced. As a consequence, the implicit jamming attack incurs additional delays at AP_2 in forwarding the packets from S_4 towards D_4. In other words, the jamming of a single client may affect the throughput of clients that are affiliated with different APs in other parts of the network.

The impact of the implicit-jamming attack: In order to demonstrate the potential impact of this attack on the performance of the network, we conduct a set of preliminary experiments on our wireless testbed (described later in section 4). In particular, we construct the WLAN scenario in Fig. 2 (left), where an AP maintains ongoing sessions with 5 clients and transmits saturated unicast traffic to all of these clients. We place a jammer 7 ft. away from one client (C). The jammer emits energy continuously at 0 dBm (1 mW), such that it causes interference to client C only. Fig. 2 depicts our throughput measurements, with and without the jammer. We observe that in the absence of jamming each client receives 4.1 Mbits/sec, on average. When the jammer is enabled, however, the long-term throughput of *all* clients drops to 90 Kbits/sec. We explain this behavior analytically in section 2.1.

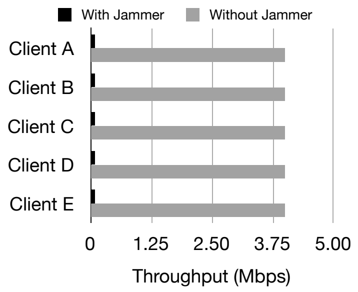

Fig. 2 The implicit jamming attack can tremendously degrade the total network throughput.

FIJI: An anti-jamming system to mitigate the implicit-jamming attack: In order to alleviate the effects of this intelligent attack, we design and implement FIJI, a distributed software system that is executed locally at the APs. With FIJI, the AP is able to quickly detect an implicit jamming attack and identify the clients that are under the direct influence of the jammer(s). Furthermore, via a minimal set of online calibrating measurements that characterize the impact of the attack, the AP shapes the downlink traffic such that: *(a)* the jammed clients receive the maximum possible throughput given the circumstances, and *(b)* the rest of the clients are unaffected, i.e., shielded from the influence of the jammer(s). Some parts of FIJI are implemented on the *Click* software framework [6] and the rest are implemented on the driver/firmware of our wireless cards. We apply FIJI on two different 802.11a/g wireless testbeds [7]. Via extensive experiments, we observe that FIJI effectively mitigates the implicit-jamming attack by *restoring the throughput to the non-jammed clients while providing the maximum possible throughput to the jammed ones.*

Our work in perspective: FIJI can be potentially applied in scenarios wherein jammers attack APs directly. However, in this work, we focus on addressing intelligent jammers that exploit the performance anomaly at the client side. Moreover, note that the impact of implicit jamming is exacerbated in downlink traffic scenarios; with uplink traffic, jammed clients will simply defer accessing the medium and will thereby allow the other clients to obtain higher levels of access.

The remainder of the paper is organized as follows. In section 2, we provide a brief background on the performance anomaly in 802.11 as well as jamming attacks, and discuss related studies. In section 3, we describe the implicit jamming detection and mitigation with FIJI, our anti-jamming system. We describe the implementation of FIJI and evaluate its effectiveness in section 4. In section 5, we elaborate on certain attributes of FIJI and discuss the applicability of our framework in different settings. We conclude in section 6.

2 Background and Previous Work

In this section, we first describe the so-called performance anomaly with IEEE 802.11 and efforts related to addressing the anomaly. We then discuss jamming attacks in brief as well as prior work related to anti-jamming.

2.1 Performance Anomaly in 802.11 wireless access networks

Heusse et al. [4] were the first to observe that the long term throughput of all the clients associated with an AP in a WLAN is limited by the client with the poorest link. This effect eventually provides the same long-term throughput to all clients. Although [4] considers uplink traffic, this “anomaly” arises with downlink traffic as well [8,9]. With either uplink or downlink saturated traffic, 802.11 provides equal medium access probability to all links. Let us consider the downlink scenario. An AP$\rightarrow$client link with low SINR will coerce the rate adaptation mechanism at the AP to use a low transmission rate for this client. Thus, when attempting to serve this client, the AP will spend large amounts of time. Given that the AP will access the channel with equal probability for low-SINR clients and high-SINR clients (higher bit rate, shorter transmission durations), the latter will be served for smaller proportions of time.

Let us assume that AP α is sending saturated unicast traffic to each of its κ clients. The *theoretical* instantaneous transmission rate from AP α towards client c_i, where $i \in \{1, ..., \kappa\}$, is a step function of the SINR for this client [10]. In this work, we consider f_{c_i} to be the instantaneous *deliverable* rate towards client c_i, which in practice may not always be equal to the transmission rate (especially at high rates). Each client

c_i of AP α will receive the ***same*** throughput T_i in the long term; this throughput is given by:

$$T_i = M_\alpha \cdot \frac{B}{\sum_{i=1}^{\kappa} \frac{B}{f_{c_i}}} = M_\alpha \cdot \frac{1}{\sum_{i=1}^{\kappa} \frac{1}{f_{c_i}}} \quad . \tag{1}$$

In the above equation, M_α is the fraction of the time that AP α is able to access the medium, given the contention with its co-channel neighbor devices. We assume that AP α transmits data packets of the same length B to all clients. From the above equation it is evident that if a client c_i receives low throughput, *all* clients will also receive equally low throughput under saturated conditions. Note that this phenomenon has been taken into account during the design of previous performance improvement algorithms for WLANs; examples can be found in [3], [8], [9], [10]. All these studies take the anomaly as a given and try to improve the network performance through other intelligent strategies, such as AP load balancing and power control. In other words, such studies are inherently based on the fact that the 802.11 MAC protocol provides long-term fairness. Clearly, when this property of 802.11 is exploited by a malicious attacker, the performance of the schemes that are based on this property is also compromised. Hence, the existence of a mechanism that detects and mitigates such jammers becomes very vital.

2.1.1 Studies on mitigating the performance anomaly in 802.11

There have been numerous efforts on addressing the anomaly in 802.11. However, most of them either require significant modifications on the 802.11 protocol functionality or they are very difficult to implement in practice.

Packet aggregation: Razafindralambo et al., [11] propose *PAS*, a technique that involves packet aggregation with dynamic time intervals. With PAS, nodes transmit consecutive packets back-to-back, separated by a SIFS period [12]. As a result, high-rate clients are able to transmit/receive many packets during an allocated time interval. However, packet aggregation requires modifications on the 802.11 protocol, in order to allow back-to-back data frame transmissions.

Contention window manipulation: Kim et al., [13] show that the anomaly can be addressed by tuning the 802.11 contention window size. They compute the minimum value of the window for the elimination of the anomaly. This technique, however, requires modification to the algorithm that selects the value of the contention window in 802.11. In contrast, our proposed scheme (described in the following section) does not require any changes to the 802.11 protocol semantics.

Data traffic manipulation: Bellavista *et al.*, in [14] propose *MUM*, an application-level middleware for facilitating multimedia streaming services. MUM tries to detect the anomaly by monitoring the RSSI of received packets and estimating the goodness of links. It employs the Linux `tc/iptables` to implement a hierarchical token buffer scheduler [15] that "differentiates" data transmissions towards low-rate nodes. The RSSI, however, cannot accurately capture the levels of contention and interference [16]. In addition, [14] uses a limited set of 4 static rate classes for traffic differentiation; *this setting is not adequate in jamming scenarios, as we show in section 4.* Along the same lines, Dunn *et al.*, [17] propose a heuristic for allocating a packet size to every client, which is proportional to the transmission rate. *We show in section 4 that the use of this heuristic during an implicit-jamming attack leads to some undesirable effects that in* *turn lead to poorer throughput than what is possible with FIJI.* Similar approaches are followed in [18,19] and [20]. Furthermore, Yang et al. [21] analytically model a WLAN with stations that support multiple transmission rates in order to demonstrate the performance anomaly. In contrast with these studies, our anti-jamming solution addresses the fact that the maximum transmission rate achieved by a single client can bound the total AP throughput. Tan and Guttag [22] propose TBR, a practical scheme that can operate in conjunction with any MAC protocol to provide long-term time-based fairness in WLANs. TBR schedules packet transmissions, taking into account the quality of the links of the AP with its clients. The goal of this scheme is to provide an equal amount of long-term channel occupancy time to each competing node. They show via experimentation that this approach can prevent faster nodes from being dragged down by slower ones. The authors provide a proof-of-concept implementation of TBR on an 802.11b small-scale testbed. However, allocating a significantly large amount of service time (order of hundreds of milliseconds) to jammed clients will have a tremedous impact to the achieved throughput of healthy clients. This is especially pronounced in cases where healthy clients run interactive applications. In addition, although they discuss the potential overheads of their implementation (due to the exchange of control messages between APs and clients), they do not quantify these overheads in 802.11a/g/n networks, which support much higher transmission rates. Note also that the main idea behind TBR resembles the operation of DRT (explained later in section 3), since the jammed client will be allocated a particular time period for being served, which may not be sufficient to sustain the maximum possible throughput that this client can achieve under jamming. We discuss TBR further in section 4. Guo et al. [23] utilize the idle communication power of wireless interfaces during TCP sessions, to improve the throughput and energy efficiency of stations in multi-rate WLANs. The idea here is that links that support transmission rates no only suffer low throughputs, but also (a) significantly degrade the performance of other nodes, and (b) have to stay awake to generate timely TCP acknowledgements, and this is energy inefficient. Given this, the authors propose a data forwarding mechanism and an energy-aware channel allocation mechanism. However, they do not consider UDP traffic, which is typically more aggressive and with which nodes need to stay awake for longer time periods. In addition, they also do not consider the presence of malicious adversaries. Finally, Bahl et al. [24] propose *SoftRepeater*, a practical system with which, clients cooperatively mitigate the 802.11 performance anomaly. SoftRepeater allows a set of clients that maintain good links qualities with their AP, to act as repeaters for the poor-link clients (which are typically located further away from the AP), in order to improve the overall network performance. However, SoftRepeater is unable to mitigate the effects of the implicit jamming attack. This is because SoftRepeater assumes that the cause of poor performance is the low SINR levels due to either distance from the AP or increased legitimate interference. In fact, this is how the implicit jamming attack can intelligently mislead legitimate nodes with regards to the reception SINR. Note that an implicit jammer can attack clients with quite high link qualities, as long as it uses a very low transmission power in way that no other devices are explicitly jammed, as we discuss in section 4. Therefore, even if packets are replayed by helper clients (as per SoftRepeater), those replayed packets will also suffer very frequent collisions due to the attack.

From the above discussion, as well as our measurements in section 4, it becomes evident that prior efforts on overcoming the performance anomaly problem in 802.11 cannot efficiently mitigate implicit jammers. We approach the 802.11 anomaly from the security point of view; in particular we examine a case where a malicious adversary can remotely exploit this feature as a vulnerability to cause complete starvation to the associated clients. *FIJI is effective against the implicit jamming attack, provides the best trade-offs between throughput and fairness and does not require any modifications on the 802.11 protocol.*

2.2 Jamming in Wireless Networks

Jammers are classified into two main categories based on their behavior.

Constant jammers emit electromagnetic energy all the time. This category includes *deceptive* jammers [25], which transmit seemingly legitimate back-to-back data packets. With this, deceptive jammers can mislead other nodes and monitoring systems into believing that legitimate traffic is being sent over the medium. Due to this, deceptive jammers are more difficult to detect. This jamming technique is not usually adopted, since it depletes the battery of mobile jammers rather quickly. Indeed, continuously emitting signals limits the ability of jammers to be autonomous and thereby their moving flexibility, since they will highly likely have to depend on external power sources.

Intermittent jammers conserve battery life by emitting energy intermittently. As examples: (i) *Random* jammers alternate between random jamming and sleeping periods. (ii) *Reactive* jammers emit energy right after the detection of traffic on the medium, and remain inactive as long as the medium is idle. The implementation of reactive jammers is difficult; however, the detection and alleviation of reactive-jamming attacks is very challenging. More details on the different jamming models can be found in [25].

A large body or prior studies employ frequency hopping (FH) techniques to overcome the presence of jammers [26–28]. Frequency hopping can be either proactive or reactive. With proactive FH, nodes periodically hop between frequencies, in order to avoid being interfered with potentially malicious jammers in their neighborhood. With reactive FH, nodes that suspect the existence of a jammer in their vicinity, switch to a new frequency and re-establish their sessions on the new frequency after appropriate handshaking. We do not employ any frequency hopping related operations in FIJI for the following reasons. First, FH cannot alleviate the influence of a wide-band jammer [29] [30], which can effectively jam all the available channels. Second, FH is ineffective in scenarios with multiple narrow-band jammers residing on every available channel. Given that there are few orthogonal channels with 802.11 (12 channels with 802.11a and just three with 802.11g), frequency hopping by itself is unlikely to be very effective. In addition, recent studies have shown that a few cleverly coordinated, narrow-band jammers can practically block the entire spectrum [31]. Third, most previously proposed FH schemes assume that the hopping between channels is fast. However this is not always the case in real systems. Specifically, FH might require a significant amount of time based on the hardware used. Prior work [32] has reported that switching between frequencies and restoring the data session may take up to 1000 msec. During the switching period there is no traffic flowing from/to the node, and this decreases the long-term throughput. However, even if such delays can be minimized with driver modifications [33], multiple jammers residing on different channels as well as wideband jammers can easily render FH inadequate. Therefore, FIJI does not rely on frequency hopping, although it could be used in conjunction.

2.2.1 Previously proposed anti-jamming techniques

Prior work has focused on the impact of jamming on the performance of isolated wireless links. To the best of our knowledge, FIJI is the first system to examine the effects of implicit jamming on the *overall* performance of WLANs and mesh networks. Gummadi *et al.* [26] find that 802.11 devices are vulnerable to specific patterns of narrow-band 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. Navda *et al.* [27] implement a proactive frequency hopping protocol with pseudo-random channel switching. They compute the optimal frequency hopping parameters, assuming that the jammer is aware of the frequency hopping procedure that is followed. Xu *et al.* [28] propose two anti jamming techniques: reactive channel surfing and spatial retreats. However, they do not consider 802.11 networks. In [25], efficient mechanisms for jammer detection at the PHY layer are developed. However, the authors do not propose any anti-jamming mechanisms. The work in [34] suggests that the proper adjustment of transmission power and error correction codes could alleviate jamming effects. However, it neither proposes an anti-jamming protocol nor performs evaluations of these strategies. Along the same lines, Lin and Noubir [35] present an analytical evaluation of the use of cryptographic interleavers with various coding schemes to improve the robustness of wireless LANs. In subsequent work, Noubir and Lin [36] investigate the power efficiency of a jammer. They show that in the absence of error-correction codes a jammer can conserve battery power by simply destroying only a portion of a legitimate packet. Xu, Trappe and Zhang [37] exploit the ability of a receiver to identify that an arrived packet is corrupted in the presence of jamming. They propose an anti-jamming technique with which, data is encoded based on the inter-arrival times between receptions of corrupted packets. They show that this results in a low-rate channel under jamming. Finally, Noubir [38] proposes a combination of directional antennae and node-mobility in order to alleviate jammers. The idea here is to achieve a high antenna gain from the transmitter to the receiver and vice versa; with this, the relative ratio of the jamming power to the signal power will be reduced. The same effect can be also achieved by using sectored antennas, or any other type of smart antennas that concentrate the energy of the beam towards a specific direction.

Various other jamming analysis, detection and mitigation schemes have been proposed in the literature [39–46]. However, *none of these efforts consider the implicit jamming attack; FIJI is the first system to address this attack.*

3 FIJI to Combat the Implicit Jamming Attack

In this section, we describe the design of our anti-jamming software system, FIJI. The high -level goal of FIJI is to detect the attack and restore the throughput on clients that are not explicitly jammed (**"healthy"** clients) as well as to maintain connectivity and provide the highest possible throughput to explicitly **"jammed"** clients. FIJI involves the co-design of two individual modules, executed at the AP: a *detection* module and a *traffic shaping* module. We have implemented the two modules in the kernel space (we provide implementation details in section 4).

3.1 Attack model

In this work, we focus on low-power deceptive jammers. In particular, we assume that the jamming device has the following properties:

– It is placed next to legitimate clients. With this, the jammer is able to distort packets destined to the jammed client(s). In addition, the jammer is constantly transmitting packets back-to-back, thereby prohibiting the jammed clients from accessing the medium.
– It operates at very low power. As discussed earlier, the jammer simply needs to explicitly affect one of the clients of the AP. By transmitting at low power the jammer can conserve energy and make the detection of the attack a challenging task.
– It is able to operate on a wide band (covering all the available channels); this makes frequency hopping techniques inappropriate.

We describe the operation of the detection and the traffic shaping modules in what follows.

3.2 Detecting the implicit-jamming attack

The purpose of this module is to make the AP capable of detecting the jammed clients. *Previous jamming detection schemes assume that the jammed node is always the one that performs the detection. However with the implicit-jamming attack, the AP needs to detect the jammed client(s) in order to prevent the throughput starvation of the healthy clients.* As an example, in [25] the jammed node performs a consistency check between the instantaneous PDR (Packet Delivery Ratio), and the RSSI (Received Signal Strength Indicator) that it measures on its antenna. If the PDR is extremely low (i.e., almost zero), while the RSSI is much higher than the CCA threshold[1], the node is considered to be jammed. With the implicit jamming attack, however, the AP does not know the RSSI value that is observed by each of its clients. Thus, the approach in [25] does not allow the AP to detect the implicit jamming attack.

Measuring the transmission delay per client: FIJI relies on measuring the data unit transmission delay $d_{c_i} = B/f_{c_i}$ of every client c_i at the AP. More specifically, the denominator of Eq. (1) is the aggregate transmission delay D_α incurred by AP α in order to serve all of its associated clients once; it is the sum of the individual d_{c_i} values, $i \in \{1, ..., \kappa\}$, of the κ clients that are associated with AP α [3]. In other words, if we assume saturated downlink traffic, D_α corresponds to the average time that AP α needs in order to send one data unit to every client. The value of D_α is the same for all clients, and the transmission delay d_{c_i} of client c_i contributes to the value of D_α. Hence, a sudden, very large increment in D_α indicates that one or more of the d_{c_i} values has suddenly increased; *this would imply that one or more clients are under attack*. Towards calculating D_α, AP α needs to measure the d_{c_i} value for every client c_i (this includes possible retransmission delays and the rate-scaling overhead[2]). Measuring d_{c_i} will directly reveal the jammed clients: the value of $d_{c_i^J}$ for a jammed

[1] The CCA (Clear Channel Assessment) threshold specifies the RSSI value below which, receptions are ignored with regards to carrier sensing [10].

[2] The rate scaling overhead accounts for the higher delays incurred due to transient lower rates that the rate adaptation algorithm invokes.

client c_i^J is likely to be much higher than the delays of the other clients. We adopt this detection strategy in FIJI.

3.3 Shaping the traffic at the AP to alleviate jammers

A trivial solution to the problem of mitigating the attack would be for the AP to simply stop serving the jammed clients. However, this would be unfair, since in many cases the jammed clients might still be able to receive data, albeit at lower rates. We opt to provide a **fair** bandwidth allocation solution; our twofold objective is to simultaneously achieve the following:

– **Objective 1:** For each of the healthy clients we seek to provide the same throughput that they would have enjoyed in the absence of the jammer, i.e., prior to the attack.
– **Objective 2:** A jammed client typically cannot receive much throughput as long as the jammer is active. Hence we want to provide to every jammed client the maximum possible throughput that it can receive, given that objective 1 is satisfied.

We refer to the state where these objectives are met as the ***optimal state***.

We propose a real-time, cross-layer software system to mitigate the effects of the implicit-jamming attack. The system is implemented partly in the Click module [6] and partly in the wireless driver/firmware. Click receives information from the MAC Layer with regards to the properties of the jammed clients. The AP$\rightarrow$client traffic is then appropriately shaped and forwarded down to the MAC layer at the AP.

3.3.1 DPT: Controlling the data packet size

With this strategy, the AP fragments the packets destined to jammed clients; each such smaller fragment is now an independent packet. We call this approach DPT for *Data Packet Tuning*. With DPT, the rate at which these smaller packets are sent to the MAC layer is equal to the rate at which normal packets were forwarded to the MAC layer, prior to jamming. DPT is expected to have the following effects: *(a)* The transmission of small data packets is more robust to interference due to jamming; hence these small packets are more likely to be correctly deciphered by the jammed clients. *(b)* The rate at which the AP accesses the medium for the jammed clients remains unchanged; however, the channel occupancy time that is spent for them is reduced, due to transmitting smaller packets to jammed clients. Hence, the AP will allocate a larger fraction of time for healthy clients. We reiterate that by maintaining the same rate of packet forwarding down to the MAC layer at the AP, we implicitly reduce the long-term data rate towards the jammed clients.

Deriving the optimal data packet sizes: Our target is to determine the right packet size such that the optimal state is reached. The problem of achieving this state is formulated as follows.

Let us suppose that AP α has κ associated clients, and that n clients are being jammed, with $n \leq \kappa$. Our objective is to *minimize the aggregate transmission delay D_α^J of all the jammed clients c_i^J, $i \in \{1,..,n\}$ of AP α*. In other words, we seek to minimize

$$D_\alpha^J = \sum_{i=1}^{n} d_{c_i^J} = \sum_{i=1}^{n} \frac{J_i}{f_{c_i^J}} \ ,$$

where J_i is the data unit length for jammed client c_i^J, while $f_{c_i^J}$ is the deliverable rate at c_i^J.

Constraint: The $d_{c_i^J}$ value of each jammed client c_i^J must be at least equal (and as close as possible) to its data unit transmission delay d_{c_i} in benign conditions:

$$X1: \quad d_{c_i}^J \geq d_{c_i} \Rightarrow \frac{J_i}{f_{c_i^J}} \geq \frac{B}{f_{c_i}},\ \forall i \in n\ ,$$

where B is the default data unit length that the AP is using for all clients, and f_{c_i} is the deliverable rate to c_i^J in benign conditions. As explained earlier, the value of D_α is the same for all clients that are associated with AP α. If we sum constraint $X1$ over all jammed clients, the left hand side of the inequality is our objective function. With this we make sure that the healthy $\kappa - n$ clients will indeed experience an aggregate transmission delay very close to $D_\alpha = \sum_{i=1}^{\kappa}(B/f_{c_i})$; note that this is the aggregate transmission delay that was experienced by these clients prior to the jamming attack. Hence, by choosing the packet size J_i that results in a transmission delay that is as close to d_{c_i} as possible, we ensure that the throughput of the healthy clients remains unaffected (we elaborate on this later with an example).

Based on the above constraint, our optimization problem can be formulated as follows:

$$\textit{minimize}: D_\alpha^J = \sum_{i=1}^{n} d_{c_i^J} = \sum_{i=1}^{n} \frac{J_i}{f_{c_i^J}} \tag{2}$$

$$\textit{subject to}: \quad 1 \leq J_i \leq B,\ \forall i \in \{1, 2, ..., n\}, \tag{3}$$

$$\textit{and}\ X1. \tag{4}$$

The solution to the above problem provides the values of J_i that minimize (2). Although the problem is an integer programming problem, its special form ensures that it always has a solution, which can be found in polynomial time w.r.t. the number of variables. We show these properties in what follows.

Lemma 1 *Our optimization problem always has a solution.*

Proof Our objective function is linear. As such, it is convex and it has a minimum in the closed and bounded set I^n, where $I = [1, B] \subseteq \Re$. Therefore, the function exhibits a minimum on the points that we retrieve, if we sample set I to pick integers in $J = \{1, 2, ..., B\}$. Hence, our objective function has a minimum on the set J^n, and on each of its non-empty subsets. We now prove that if we take into account the constraint $X1$, a feasible point still exists (i.e., the constraint set is not empty). In other words, we show that there is a non-empty subset of J^n that satisfies the constraint $X1$. We have $f_{c_i^J} \leq f_{c_i}$, since the deliverable rate for the jammed client is expected to be at most what it was in benign conditions. We now consider the following two cases for the relation between $f_{c_i^J}$ and f_{c_i}.

– If $f_{c_i^J} = f_{c_i}$, then $\frac{1}{f_{c_i^J}} = \frac{1}{f_{c_i}}$. If we select $J_i = B\ \forall i$, we conclude that constraint $X1$ is satisfied.
– If $f_{c_i^J} < f_{c_i}$, then $\frac{1}{f_{c_i^J}} > \frac{1}{f_{c_i}}$ (since both quantities are positive). Let us pick $J_i \geq B \cdot \frac{f_{c_i^J}}{f_{c_i}}$. The right hand side of the inequality is smaller than B since $\frac{f_{c_i^J}}{f_{c_i}} < 1$. Thus, there exists such a J_i, and: $\frac{J_i}{f_{c_i^J}} \geq \frac{B}{f_{c_i}}$. Hence constraint $X1$ is satisfied.

Consequently, our problem always has a solution.

Lemma 2 *Our optimization problem can be solved in polynomial time w.r.t. the number of variables.*

Proof The value of our objective function increases as we increase the value of each variable J_i. As a result in order to minimize the objective function we need to keep these J_is as small as possible. Constraint X1 defines the smallest possible real value for each one of the variables. Since J_i can take only integer values we just need to pick the smallest possible integer that satisfies constraint X1. This needs $O(1)$ time for every variable and given that we have n such variables our optimization problem can be solved in linear time, $O(n)$.

How does DPT operate? Let us consider a case study with AP α, $\kappa = 3$, $n = 1$ and default packet size B. The transmission delays for the healthy clients c_1 and c_2 are d_1 and d_2, respectively; for the jammed client c_3, it is d_3. The long-term throughput of every client in benign conditions will be:

$$T_b = \frac{B}{d_1 + d_2 + d_3}.$$

If c_3 is now being jammed, its transmission delay will be $d_3^J > d_3$ and the new throughput will be:

$$T_J = \frac{B}{d_1 + d_2 + d_3^J}.$$

By applying DPT, the packet size towards c_3 will be J_3^{dpt} and its new transmission delay will be d_3^{dpt}. Since the rest of the clients are to maintain their old transmission delays (they are not explicitly jammed), the throughput with DPT will be:

$$T_{dpt} = \frac{B}{d_1 + d_2 + d_3^{dpt}}.$$

Our minimization problem ensures that $d_3^{dpt} \approx d_3$. Thus, for clients c_1 and c_2:

$$T_{dpt_1} = T_{dpt_2} \approx T_b.$$

In other words, DPT restores the throughput at the healthy clients.

Next, we show that the jammed client cannot receive a higher throughput if we further decrease the packet size[3] to a value $J_3^l < J_3^{dpt}$. With packet size J_3^{dpt} the throughput at c_3 will be:

$$T_{dpt_3} = \frac{J_3^{dpt}}{d_1 + d_2 + d_3^{dpt}}.$$

Let us assume that with packet size $J_3^l < J_3^{dpt}$ the transmission delay of c_3 is d_3^l. The throughput at c_3 will then be

$$T_{l_3} = \frac{J_3^l}{d_1 + d_2 + d_3^l}.$$

[3] For larger packet sizes, objective 1 cannot be satisfied; hence we do not need to consider such a case.

The required condition $T_{l_3} < T_{dpt_3}$ can be simplified as:

$$T_{l_3} < T_{dpt_3} \Leftrightarrow d_3^l > \frac{J_3^l}{J_3^{dpt}} \cdot (d_1 + d_2 + d_3^{dpt}) - d_1 - d_2.$$

Since the packet delivery rate f_{c_3} is the same, we have:

$$\frac{J_3^l}{J_3^{dpt}} = \frac{d_3^l}{d_3^{dpt}} \Leftrightarrow d_3^l = d_3^{dpt} \cdot \frac{J_3^l}{J_3^{dpt}}.$$

Thus: $\frac{J_3^l}{J_3^{dpt}} \cdot d_3^{dpt} > \frac{J_3^l}{J_3^{dpt}} \cdot (d_1 + d_2 + d_3^{dpt}) - d_1 - d_2 \Leftrightarrow 0 > (\frac{J_3^l}{J_3^{dpt}} - 1)(d_1 + d_2).$

The last inequality is always true; hence, $T_{l_3} < T_{dpt_3}$.

Similar steps can be followed in order to show that DPT operates in the same manner in scenarios with multiple jammed clients. We adopt DPT in FIJI.

3.3.2 DRT: An alternate approach

An alternative strategy would be to *explicitly tune the rate at which the packets are delivered* at the MAC layer (the packet size is now kept unchanged), destined to jammed clients. Fewer packets would arrive at the MAC layer for transmission towards the jammed clients, thereby allowing the AP to send traffic to healthy clients more frequently. Let us call this approach DRT for *Data Rate Tuning.* DRT operates as follows. Based on the measured d_{c_i} for each client c_i, the deliverable rate to every jammed client would be:

$$f_{c_i^J} = \frac{B}{d_{c_i^J}}. \tag{5}$$

DRT would bound the packet generation rate such that the data rate to the jammed client c_i^J is at most $f_{c_i^J}$. As a result, the rest of the (healthy) clients would share the remaining bandwidth. Thus, they would enjoy a share that is in fact higher than what they had prior to the attack. However, the packets destined to the jammed clients could be potentially lost due to channel or interference effects. Hence with DRT, the jammed clients will eventually receive *lower long-term throughput* than the specified (by DRT) rate of $f_{c_i^J}$. Clearly, while both DPT and DRT shape the traffic in order to overcome the implicit jamming effects, they essentially differ in the way they allocate the bandwidth. With DPT the healthy clients receive the *same throughput as before the attack*, while the jammed clients achieve the *maximum possible* throughput under the circumstances. On the other hand, with DRT the healthy clients have a higher share of the bandwidth than in benign settings and receive *more throughput than before the attack*; the APs will spend more time serving the healthy clients, since most of the traffic is now destined to them. However, since the jammed clients do not reach their capacity, they are treated rather "unfairly". We evaluate this fairness versus throughput trade-off in section 4.

4 Implementation and Evaluation

In this section, we first describe our implementation of FIJI. Next we apply FIJI on two experimental testbeds and evaluate its efficacy in overcoming the implicit jamming attack.

4.1 The implementation of FIJI

FIJI is implemented entirely at the AP; no client software modifications are needed. In addition, FIJI does not require any special functionalities at the APs or at the clients; the only requirement is for the AP to be able to measure the d_{c_i} value for each affiliated client. Hence, FIJI can be easily applied on commercial APs through a minor driver/firmware update. In order to implement the two modules of FIJI we perform modifications on the driver and firmware of the AP, and we develop specific traffic shaping functionalities on the Click framework [6].

Implementing the implicit-jamming detection module: As explained in section 3.2, the AP needs to measure d_{c_i} for every client c_i. This will reveal, with high probability, the set of jammed clients. However, the value of d_{c_i} cannot be directly obtained from the driver of the wireless card; modifications in the firmware are required in order to compute this value. We use a prototype version of the *Intel ipw2200* AP driver/firmware; for every client we measure the time duration between the placement of the packet at the head of the MAC queue until an 802.11 ACK frame is received for this packet. The value is then passed up to the driver. The AP maintains a table in the driver space with the d_{c_i} value for every client c_i. It also computes D^J_α (when jammers are active) and D_α (when jammers are inactive), by summing up the corresponding client delays. Temporary variations of the d_{c_i} values are handled by FIJI by using weighted moving average filtering; the previously maintained average is assigned a weight of 0.9 while the new sample has an associated weight of 0.1 (similar values are used in [3,8]). Using these values, the AP constructs a table with the appropriate data packet sizes for the jammed clients. If the weighted $d_{c_i(new)}/d_{c_i(old)}$ value (for one or more clients) exceeds a pre-specified threshold δ, the AP computes the new packet sizes, updates the table and subsequently feeds it into the traffic shaping module, described below.

Implementation of the traffic shaping module: We implement the traffic shaper in Click. The module receives the table from the driver with suggested parameter settings for every client and shapes the traffic accordingly. We implement both DPT and DRT for comparison purposes. For DPT we have also developed an application-level script, which reads the table with the suggested packet sizes and inputs these values to the rude/crude measurement tool [47]. For DRT one may use two different Click elements, namely either the `BandwidthShaper(bandwidth)` or the `LinkUnqueue(latency, bandwidth)` element; we utilize the latter. Finally, we configure the AP to periodically flush the stored transmission delay values for every client and perform fresh delay measurements, using the default packet size. With this, we address scenarios of mobile jammers, which may move to the proximity of different clients, jammers with variable transmission power as well as jammers that stop operating.

4.2 Experimental set-up and methodology

Description of the experimental networks: We use two different wireless testbeds (we refer to them as testbed-A and testbed-B respectively) throughout the evaluation of FIJI [7]. Testbed-A consists of 28 Soekris net4826 nodes [48], which mount a Debian Linux distribution with kernel v2.6 over NFS. The testbed is deployed in the 3rd floor, at the research labs wing of Engineering Building Unit II, at the University of California, Riverside. The node layout is depicted in Fig. 3a. Each node is equipped with an *Intel-2915* and a Wistron Neweb CM9 mini PCI card, which carries the AR5213 as the main chip. Each card is connected to two 5-dBi gain external omnidirectional antennae. We use both the *main* and *aux* antenna connectors of the card for diversity. As mentioned earlier, we use a proprietary version of the *ipw2200* AP driver/firmware of the *Intel-2915* card. With this version we are able to (a) measure the D_α and D_α^J values at the AP, and (b) experiment with both 802.11a and 802.11g. We also use the Madwifi-ng driver for the Atheros-based cards. We utilize testbed-A to evaluate FIJI in WLAN scenarios with UDP traffic. Testbed-B consists of 11 nodes, it is deployed in the same building at the faculty wing, and has the exact same hardware and software configuration as testbed-A. However, the deployment area is quite different, since nodes belonging to testbed-B have been deployed in small rooms, and there is no line of sight between them. The deployment of testbed-B is shown in Fig. 3b.

The different properties of the two testbeds, in terms of deployment, enable us examine the effectiveness of FIJI in diverse topologies. Meanwhile this ensures that the observed relative performance (with each testbed) is not affected by the hardware and the software capabilities of the individual devices.

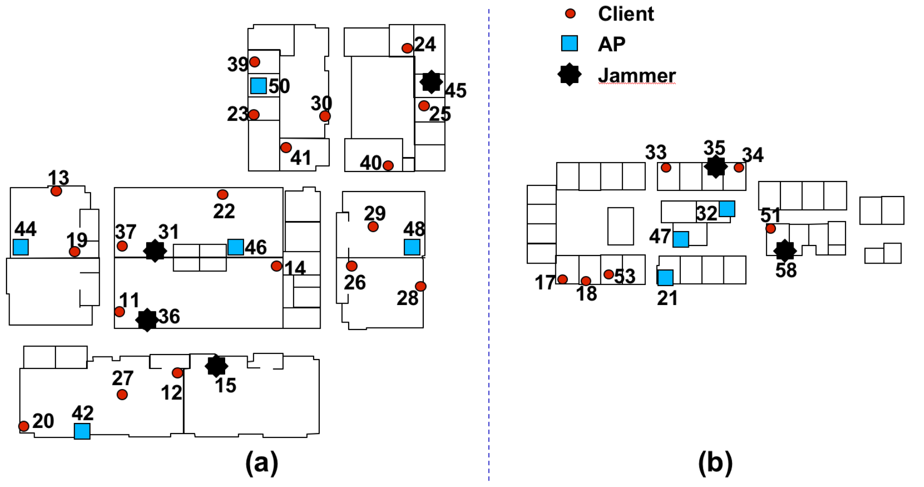

Fig. 3 The deployment of our 802.11a/g testbeds in the 3rd floor of our campus building. The clients are represented by dots, the APs by squares and the jammers by stars.

Constant jammer implementation: We have implemented our own deceptive jammer (instead of purchasing a commercial one [2]) since this gives us the freedom of tuning various jamming parameters. Our implementation of a constant jammer is based on a specific card configuration and a user space utility that sends broadcast packets as fast as possible. Our jammers are also equipped with the Intel-2915 cards; our ipw2200 prototype firmware for these cards allows the tuning of the CCA threshold parameter. By setting the CCA threshold to 0 dBm, we force the WiFi card to ignore all 802.11 signals during carrier sensing (packets arrive at the jammer's circuitry with powers much less than 0 dBm, even if the distances between the jammer and the legitimate transceivers are very small). 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 (the back-off functionality is disabled in 802.11 for broadcast traffic). We have developed an application-layer utility that employs *raw sockets*, allowing the construction of UDP packets and the forwarding of each packet directly down to the hardware.

Experimental methodology: For each experiment we first enable traffic from the AP to its clients and subsequently we activate the jammer(s). The duration of each experiment is 10 minutes; during each minute, the jammer is inactive for the first k sec, where $k \in [5, 20]$, and active for the other $60 - k$ sec. We use a subset of 6 nodes as the jamming devices (nodes 15, 31, 35, 36, 45 and 58 in Fig. 3). We collect throughput and transmission delay (d_{c_i}) measurements once every 500 msec, for each client. We experiment with many different topological settings, with different numbers of APs and clients. By default all legitimate nodes set their transmission powers to the maximum value of 20 dBm and their CCA thresholds to -80 dBm. We examine both 802.11a and 802.11g links (unless otherwise stated, we observe the same behavior for 802.11a and 802.11g). The experiments are performed late at night in order to avoid interference from collocated WLANs, as well as not to cause interference to them. We use saturated UDP traffic with a default data packet size $B = 1500$ bytes. We also experiment with TCP traffic. We use the *iperf* measurement tool to generate data traffic among legitimate nodes. We also use the *rude* tool to test DPT.

In what follows, we apply our anti-jamming framework on our testbeds and evaluate its efficiency in alleviating the effects of implicit-jamming on the network performance. We first discuss our experiments on testbed-A, where we consider WLAN settings and UDP traffic. Subsequently we discuss our experiments on testbed-B, where we evaluate FIJI on a mesh topology with both UDP and TCP traffic.

4.3 Assessing the efficacy of FIJI in WLAN settings

Next we evaluate FIJI in WLAN deployments, where packets travel at most 1 hop (i.e., from the APs to their clients).

4.3.1 The efficacy of the detection module

We seek to observe two properties of this module:

1. *Efficiency of Detection*: How quickly can FIJI detect the presence of implicit jammers?
2. *Accuracy of Detection*: How accurately can FIJI determine if there is an ongoing jamming attack?

We conduct experiments with 5 APs and different numbers of clients with various link qualities. We configure the jammers to transmit at 0 dBm (1 mW) with CCA = 0 dBm, such that they affect one or more clients without affecting the APs.

a) On the speed of detection: Our measurements indicate that the transmission delay $d_{c_i^J}$ of a client increases sharply upon experiencing the implicit jamming attack. This increase is seen in less than 700 msecs; this time includes the transient periods before the weighted average $d_{c_i^J}$ converges to a stable value. Fig. 4 depicts a delay snapshot with one AP and four clients with moderate-quality links. We observe that the $d_{c_1^J}$ value increases significantly (by 26 times in this experiment). Other experiments

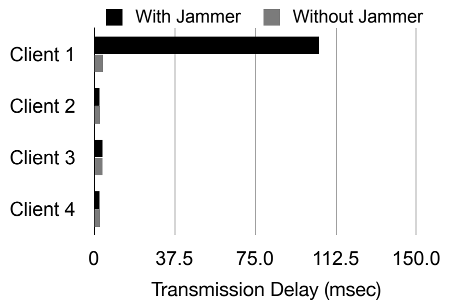

Fig. 4 FIJI detects jammed clients by measuring their data unit transmission delays.

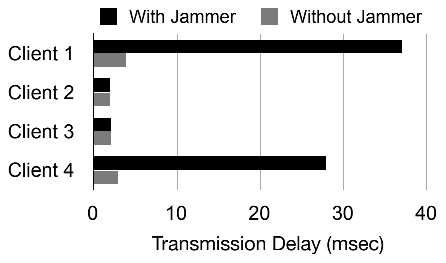

Fig. 5 The jammer detection functionality of FIJI is accurate in most cases.

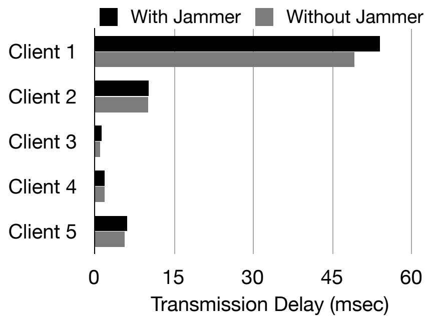

Fig. 6 The jammer detection with FIJI is less accurate in scenarios with very poor links.

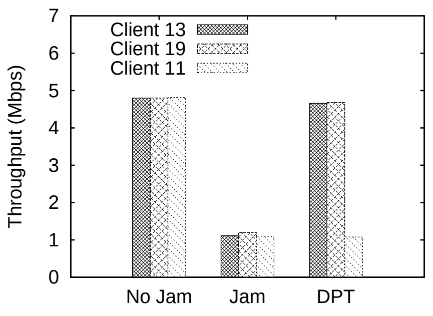

Fig. 7 DPT restores the performance of healthy clients to that in benign settings.

provided similar results. In summary, these results show that FIJI can quickly detect implicit jamming attacks.

b) On the accuracy of detection: We seek to evaluate FIJI in terms of its ability to detect an implicit jamming attack in the presence of interference. Note that the d_{c_i} value for a client c_i is affected by the levels of interference on the AP $\rightarrow c_i$ link. The higher the level of interference, the higher the d_{c_i} value. In order to evaluate this ability of FIJI, we perform experiments with multiple overlapping cells (each with its own AP), so that some clients suffer interference from one or more APs; in this setting, we activate our low-power jammers.

Detecting jamming on good quality links: We first consider links that have a high SINR. Fig. 5 depicts sample experimental results. In the snapshot of Fig. 5, a jammer is placed such that it affects 2 out of the 4 clients of an AP. We observe that FIJI is able to perform a successful detection. In general, our empirical observations suggest that when threshold $\delta \geq 9$, FIJI can effectively detect the attack (Fig. 5). In the experiment described above, the value of δ was 9.

FIJI and poor quality links: With poor quality links (SINR is low), FIJI cannot easily decide if a client is under attack or not. This effect is captured in Fig. 6, where the jammer affects a very poor link. In particular, the link 46→25 is considered with the node 45 acting as a jammer (Fig. 3). The link achieves 190 Kbits/sec in the absence of jamming and 164 Kbits/sec under jamming. Since the jammer does not significantly increase the delay experienced on such poor links, FIJI cannot decipher whether the increased $d_{node-25}$ value is due to jamming or legitimate interference. However, in such conditions, the overall change in the network performance due to the jammer is unlikely to be significant; the presence of the poor link already hurts the network performance. Furthermore note that a jammer is unlikely to attack such poor quality links if it aims to harm the network to the extent possible.

In some extreme cases, a poor quality link (exposed perhaps to other interfering APs that are hidden from its own AP) might cause a client to experience large delays. In such scenarios with healthy but poor-quality links, FIJI may incorrectly classify such links as being *jammed*. Classifying such cases as attacks, though, is perhaps appealing in terms of improving performance for the rest of the network.

FIJI and high power jammers: An implicit-jamming attacker is likely to place its jammer(s) very close to one or more clients so as to:

- degrade the client's observed SINR value to the extent possible, and
- use a very low transmission power, in order to conserve energy and avoid detection.

As our experiments indicate, under these conditions, FIJI can identify the jammed clients in real time since all measured $d_{c_i^J}$ values are usually extremely high for those clients. In contrast, a jammer could use high transmission power (although this could increase the chance of its detection and result in high energy consumption). Such a high power jammer is likely to affect multiple clients and even the AP itself, directly. The delays of all these clients may go up and in this case, given its design principles, FIJI may not be able to detect the jammer. However, there are other jammer detection techniques that can be used in conjunction with FIJI to detect such jammers [25].

4.3.2 The traffic shaping module in action

Next we evaluate the efficacy of DPT and compare it against DRT.

DPT is the most fair solution: In a nutshell we observe that as long as the jammer is successfully detected, DPT restores the throughput at the healthy clients. A sample case is depicted in Fig. 7. Here, AP 44 transmits unicast traffic to clients 11, 13 and 19; node 36 is jamming client 11. In the absence of jamming each client receives 4.8 Mbits/sec on average. When the jammer is active, without enabling DPT, all clients receive 1.1 Mbits/sec on average. The solution to the problem formulated in (2) suggests that J_{11} should be set to 345 bytes. When DPT is enabled and this packet size is chosen for the jammed client, we observe that the throughput of the healthy clients 13 and 19 is restored to 4.66 Mbits/sec, while the jammed client 11 achieves about 1.1 Mbits/sec. Note that the healthy clients do not achieve their jamming-free throughput of 4.8 Mbits/sec. This is because in our solution the equality in the constraint $X1$ is achieved for a non-integral value of J_{11}; we round the value of J_{11} up to the nearest integer. With this, the transmission delay for the jammed client is a bit higher as compared to the delay under benign conditions and this slightly degrades the throughput at the healthy clients.

In order to validate that DPT provides the most fair bandwidth allocation, we experiment with many different J_{11} values. Fig. 8 depicts the results that correspond to the settings with two J_{11} values: 166 and 700 bytes. We observe that:

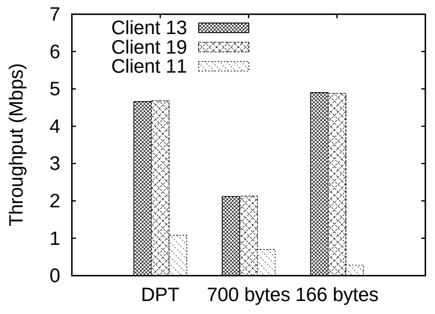

Fig. 8 DPT always manages to provide a fair allocation of throughput among clients .

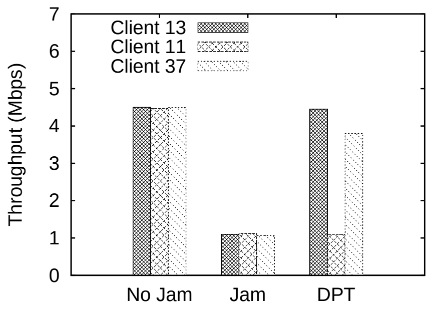

Fig. 9 DPT can easily handle scenarios with multiple clients that are simultaneously jammed.

– With packet sizes smaller than J_{11}^{dpt} (case with 166 bytes), the jammed client does not reach its capacity (receives 360 Kbits/sec) and the AP spends more time serving the healthy clients (as discussed in section 3): each healthy client now receives 5.1 Mbits/sec. Note that the value $J_{11} = 166$ bytes is computed using the approach proposed in [17] for the considered scenario and it clearly does not provide the desirable fairness in terms of throughput. Hence, while with [17] healthy clients potentially receive higher throughputs than in benign settings, jammed clients are not given the opportunity to obtain the maximum possible throughput under the circumstances.
– When the packet size is higher than J_{11}^{dpt} (case with 700 bytes), the throughput at the jammed client is lower than 1.1 Mbits/sec; the healthy clients also underperform. This is again conformant with our analytical assessments in section 3 with regards to the maximum achievable throughput.

We would like to point out here that "desirable fairness" refers to the fairness conditions described in section 3; this is where healthy clients receive as much throughput as before the jamming attack (or in the absence of implicit jammers), while jammed clients receive as much throughput as possible (although not as much as healthy clients, since they are within the direct range of the jammer). We consider this to be *fair*, since the jammer does not explicitly affect healthy clients; the throughput of such clients should not be affected by the existence of a jammer which does not directly reach them. We do not argue that the fairness strategy of FIJI is better than other potential fairness strategies (such as the one in [17]); instead we call for for deployments that consider our fairness strategy. FIJI addresses our performance objectives in a satisfactory manner. Clearly, if different performance objectives are considered, solutions such as [17] and [14] may fit better than FIJI.

As an example, if the objective is to achieve a proportional fairness based rate allocation (which is not inherently the case with the 802.11 protocol), allocating rates to links that can infact support them seems to be a more natural choice. As another example, a different criterion could consider that it is fair to allocate the same amount of time for data transmissions to all clients (e.g. see TBR [22]). While such an objective would potentially make sense in a benign setting, it is our belief that it is inappropriate in the considered setting. This is because with such a method, a jammed client would be allocated the same amount of time for data transmissions as a healthy client. We identify three possible outcomes with this approach:

1. *The jammed client is allocated a significant amount of time:* In this case, a healthy client needs to wait for a considerable amount of time until the jammed client finishes with its transmissions. This would result in very low long-term throughput for healthy clients.
2. *The jammed client is allocated an extremely small amount of time:* In such a case, the jammed client would potentially not be able to receive the maximum possible throughput in the allocated time, while a healthy client could receive higher throughput than in benign settings.
3. *The jammed client is allocated exactly the time required to achieve maximum throughput:* With this, the same outcome as with FIJI could be potentially achieved. However, given that TBR fixes the transmission time, it cannot accommodate scenarios with mobile jammers, or generally with variable channel qualities over time.

Design and analysis of solutions that consider different performance objectives (like the one described in section 3) are beyond the scope of this paper.

Multiple jammed clients: We have so far considered scenarios wherein a single client was jammed. Next, we examine scenarios with multiple jammed clients per AP. Our experiments reveal that DPT is also able to effectively mitigate the implicit jamming attack in such scenarios. Fig. 9 presents a sample case with AP 46 and clients 11, 37 and 14; the jammer-node 36 explicitly affects both clients 11 and 37. Under benign conditions all clients receive approximately 4.5 Mbits/sec on average. As soon as the jammer is activated, without enabling DPT, all clients receive about 1.1 Mbits/sec. DPT sets the value of J_{11} to be 367 bytes and J_{37} to be 1266 bytes. With this, *DPT is able to restore the throughput at the healthy clients.*

DPT vs. DRT: Using the same methodology, we examine the effectiveness of the DRT solution. Our measurements demonstrate that DRT provides much higher throughput to healthy clients. On the other hand, DRT results in an additional unfair degradation at the jammed client. Fig. 10 represents the behaviors in an example scenario, with the same topological configuration as before (AP 44, clients 11, 13 and 19, jammer 36); the figure depicts the throughput prior to the attack (benign settings), with the jammer without DRT, and after the application of DRT. We observe that DRT overcomes the implicit impacts of the attack. Upon enabling DRT, clients 13 and 19 are no longer affected by the jammer and they receive 5.12 Mbits/sec each. Although DRT sets the maximum allowable data rate towards client 11 to be 1.1 Mbits/sec, the observed throughput at this client is significantly lower i.e., 680 Kbits/sec on average. This behavior of DRT conforms with our discussion in section 3.3; we observe similar trends in all our measurements with one or more jammed clients. *To summarize, with DRT the healthy clients receive more throughput than before the attack; however the jammed clients are penalized further.*

The choice between DPT and DRT depends on the performance objectives; one has to decide between fairness (with DPT) and bandwidth utilization (with DRT). DPT is fair: the healthy clients receive the same throughput as before the attack, while the jammed clients achieve the maximum possible throughput under the circumstances. On the other hand, DRT increases the throughput at the healthy clients and potentially, the total network throughput. However, the jammed clients cannot receive the maximum throughput that they can achieve in the presence of the jammer.

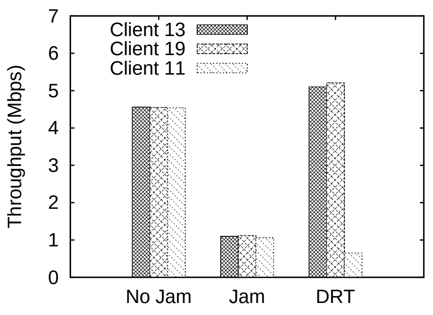

Fig. 10 With DRT healthy clients receive more throughput than before the attack.

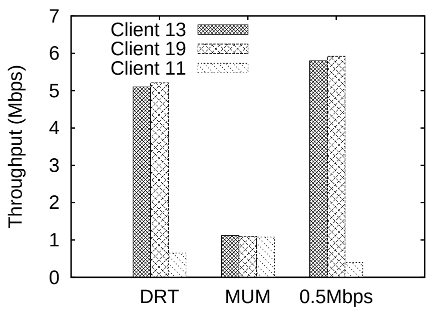

Fig. 11 DRT satisfies our objectives better than other data rate allocation approaches.

Note that DRT also relies on the online measurement and use of d_{c_i}. With this, DRT seeks to eliminate the effects of implicit jamming at healthy clients, while at the same time not degrade the throughput at jammed clients. Fig. 11 depicts a case with 802.11a where DRT sets the data rate at 1.1 Mbits/sec, while MUM [14] (recall our discussion in section 2) sets 6 Mbits/sec. We observe that by using data rates higher than the one chosen by DRT, the healthy clients are still affected by the attack, since in this case the downlink traffic for the jammed client is still saturated. Moreover, if we use lower data rates than the one chosen by DRT, the healthy clients get more service time, however the jammed clients receive much lower throughput than with DRT. More specifically, as discussed in section 2, MUM uses a limited set of static rate classes for traffic differentiation; e.g. 4 rate classes in the case of 802.11b: 1, 2, 5.5 and 11 Mbits/sec, with decreasing priorities from higher to lower rates. Clearly for 802.11a/g the set of rate classes could be increased (e.g. to 8), but again it would still be a limited set of fixed rates. Moreover, even if this rate set is further sub-divided down to more sub-sets, as per the hierarchical token buffer (HTB) concept, an extended, yet fixed rate set would be formed. This suggests that MUM could potentially be an efficient countermeasure against implicit jamming only in cases where jammed clients are assigned a data rate which "luckily" happens to be the *exact maximum* rate that the jammed client is able to receive. Stated otherwise, it is very difficult for MUM to accommodate a scenario where the jammed client would receive the maximum possible throughput, since this would require that MUM configures its HTB in such a way that the capacity of a specific buffer in the hierarchy is exactly equal to the maximum throughput that the jammed client can potentially receive. Even if such a buffer exists, the HTB configuration cannot guarrantee that healthy clients would receive throughput in a fair manner (as per our fairness criterion discussed in section 3). Allocating the jammed client's link with the AP to a buffer of higher capacity would compromise the fairness in throughput for healthy clients. Analogously, allocating the jammed client to a buffer with a capacity which is lower than the maximum possible throughput under jamming, clearly violates our objective of allowing the jammed client to receive the maximum possible throughput under the circumstances. Note that an HTB based approach such as MUM could potentially result in a per-link data rate allocation that offers the maximum total AP (or network) throughput. However, such a rate allocation does not necessarily result in a fair throughput allocation for healthy clients, nor in the maximum possible throughput for the jammed clients. In other words, while MUM could potentially offer a total AP throughput that is higher than that with DRT, it cannot always satisfy the objective postulated in section 3.

4.4 Evaluating FIJI in wireless mesh settings with TCP traffic

Previous studies have shown that the 802.11 performance anomaly exists in scenarios with TCP traffic as well [4]. In our last set of experiments, we assess the efficiency of FIJI in mesh settings with TCP traffic. Throughout these measurements[4] we select topologies wherein all legitimate links are of similar quality in terms of packet delivery ratio. With this, we are able to observe the efficacy of FIJI while ensuring that the decisions of TCP for every flow in benign conditions are similar and are not affected by legitimate interference.

As a representative example, we consider the following topology:

- Clients 17, 18 and 53 are affiliated with AP 21; clients 33, 34 and 51 are affiliated with AP 32 (see Fig. 3).
- Jammers 35 and 58 are placed close to clients 34 and 51 respectively; the jammers set their transmission powers to 0 dBm, such that they explicitly affect only clients 34 and 51.

We enable the following TCP flows between end-clients: 17$\rightsquigarrow$33, 18$\rightsquigarrow$34 and 53$\rightsquigarrow$51. Packets from the clients of AP 21 are routed through AP 47 to AP 32. The latter delivers the packets to their final destinations. The application data packet generation rate is such that the source nodes always have packets to transmit.

Traffic shaping with TCP projects the same behavior as DRT: Our measurements with TCP traffic suggest that the data rate decisions made by TCP provide a high throughput at healthy clients, while jammed clients suffer extremely low throughputs. This is due to the fact that the jammer causes frequent collisions at jammed clients and thus, TCP is coerced to drastically decrease the contention windows at the corresponding packet senders. As a result: (a) Fewer packets that are destined towards jammed clients travel through the mesh route. (b) APs deliver packets towards healthy clients for most of the time. This is depicted in Fig. 12. We observe that the throughput at node 51 (the jammed client) is practically nullified, while healthy clients (node 33 and 34) receive a much higher throughput than before the attack. The achieved performance is similar to the one achieved with DRT (recall our earlier discussion). This is somewhat expected, since both TCP and DRT react by reducing the data rate at source nodes. The same behavior is observed for the case of two active jammers (Fig. 13). These experiments show that the application of TCP cannot provide a fair throughput allocation under the presence of the implicit jamming attack.

FIJI provides a fair restoration of throughput: As with UDP traffic, the application of FIJI restores the benign throughput at healthy clients. Moreover, jammed clients achieve much higher throughputs than in the absence of FIJI (improvements range between 170% and 300% throughout our experiments).

[4] We consider TCP-Reno in this set of experiments.

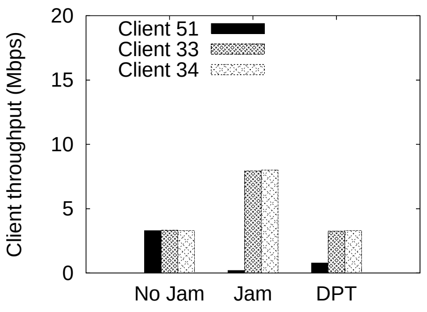

Fig. 12 FIJI provides a fair throughput allocation in mesh deployments with TCP traffic.

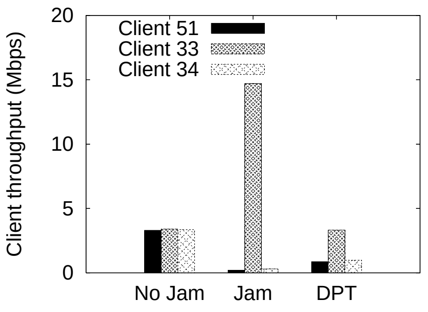

Fig. 13 FIJI allocates the throughput fairly in scenarios with TCP traffic and with multiple jammed clients.

5 Further Discussion

FIJI and previous studies on traffic shaping: Our work is the first to analytically derive the *optimal* settings for traffic shaping at the AP to mitigate the implicit-jamming attack. Traffic shapers have also been previously proposed in [14,18,19,17]. Clearly, FIJI could also be considered as another traffic shaper, simply to overcome the performance degradation due to the 802.11 anomaly. Unlike FIJI however, previous traffic shaping schemes cannot overcome the effects of an implicit-jamming attack, as explained in sections 2 and 4. Other schemes that provide fair access to the WLAN resources [49,3] would also be inadequate in combating an implicit-jamming attack since they are not designed for this purpose.

On the assumption of equal packet size in benign conditions: The scope of our work lies within the existence of the performance anomaly problem in 802.11 WLANs: under conditions with fully saturated traffic, the AP throughput is dependent on the client with the worst AP-client link quality. The performance anomaly problem is more prominent in scenarios where the AP sends packets of equal size to each associated client. Such traffic conditions are especially motivating for the application of the implicit jamming attack, since it is sufficient to simply jam a single client in order to significantly worsen the cell throughput performance. Clearly, if the attacked client sends/receives small packets, or if the corresponding link between AP and jammed client does not facilitate saturated traffic, then (a) the performance anomaly is not so prominent, and (b) the jammer has simply selected the "wrong target". We would also like to point out that FIJI is especially applicable in scenarios where all clients run the same, traffic-demanding application, such as video streaming, where the application provides packets of very large size to the lower layers, which are further fragmented by the transport layer and the MAC layer. Such fragments correspond to packets of size B, as discussed in our paper. Finally we would like to point out that the assumption of equal packet sizes in benign conditions helps rendering the problem analycally tractable and easier to understand; such assumption has been extensively used in a plurality of prior studies (such as [8], [9], etc) for the sake of analytical tractability.

FIJI versus power control: Power control has been suggested as a means of mitigating legitimate interference [9,50]. Typically with power control, nodes tune their transmission power and CCA settings in order to reduce the amount of interference from/to their neighbors. However, if the jammer is very close to one or more clients, its signal cannot be ignored through CCA adaptation. If a client increases its CCA threshold to a high level (to ignore the jammer's signal), the connectivity to the AP will be lost.

Addressing random and reactive jammers: FIJI can mitigate the interference due to any type of jammer, even random or reactive jammers. With prolonged random jamming and sleeping periods (order of seconds), FIJI can perform a rapid detection and then customize the data packet size, as per the observed data unit transmission delay $d_{c_i^J}$. If the sleep and active periods of the random jammer are of the order of milliseconds, FIJI can monitor the *average* $d_{c_i^J}$ value instead. FIJI is expected to alleviate reactive jammers, too, since it only needs to monitor the impact of reactive jamming by measuring $d_{c_i^J}$. We have not experimented with reactive jammers, since implementing such a jammer is a very difficult task.

FIJI against other attacks: The two modules of FIJI can arguably be effective against any attempt to exploit the 802.11 performance anomaly in order to degrade the client throughput. As examples, a *compromised* device x could *deliberately* decide to *(a)* associate to a very distant AP α, or *(b)* accept traffic at a very low reception rate only (e.g. by discarding a large volume of correctly received packets). In both cases, x would receive a few Kbits/sec. Note here that, legitimate, non-compromised devices would follow such an approach only if they cannot associate with a better APs. However, given that (a) dense deployments of WLANs make the presence of an AP with a good quality link likely [9], and (b) distant poor quality APs are likely to be beyond the administrative domain of the client (the client will not be able to associate with such APs), the possibility of this is small in practice. FIJI can arguably be effective against such attacks. In particular, FIJI considers such clients to be jammed clients and ensures that the other clients remain unaffected.

6 Conclusion

In this paper we identify a low-power jamming attack that we call the *implicit jamming attack*. With this attack, a jammer exploits a performance trait of the IEEE 802.11 MAC protocol to cause starvation to not only an explicitly jammed client, but all the clients associated with the same AP as that client. Since the 802.11 MAC provides long term fairness (under saturation conditions) to the associated clients in terms of equal throughput, the attacker can nullify the AP throughput by affecting only one or at most a few clients.

We design, implement and evaluate FIJI, a cross layer software system for mitigating the implicit-jamming attack. FIJI is comprised of two modules, for detecting such an attack and shaping the traffic appropriately in order to alleviate the jamming effects. We evaluate FIJI on an 802.11a/g testbed, and under many different jamming scenarios. We show that FIJI can quickly detect the attack and effectively restore the throughput at the implicitly affected clients. FIJI also ensures that the jammed clients get as much throughput as they can under the circumstances.

Acknowledgment: We thank Intel Research for providing the wireless driver.

References

1. SESP jammers. http://www.sesp.com/.

2. ISM wideband jammers. http://69.6.206.229/e-commerce-solutions-catalog1.0.4.html.
3. K. Sundaresan and K. Papagiannaki. The Need for Cross-Layer Information in Access Point Selection Algorithms. In *ACM IMC*, 2006.
4. M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance Anomaly of 802.11b. In *IEEE INFOCOM*, 2003.
5. G. Athanasiou, T. Korakis, O. Ercetin, and L. Tassiulas. Dynamic Cross-Layer Association in 802.11-based Mesh Networks. In *IEEE INFOCOM*, 2007.
6. Click web page. http://read.cs.ucla.edu/click/.
7. UCR Wireless Testbed. http://networks.cs.ucr.edu/testbed.
8. B. Kauffmann et al. Measurement-Based Self Organization of Interfering 802.11 Wireless Access Networks. In *IEEE INFOCOM*, 2007.
9. I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. Mhatre. MDG: Measurement-Driven Guidelines for 802.11 WLAN Design. In *ACM MOBICOM*, 2007.
10. V. Mhatre, K. Papagiannaki, and F. Baccelli. Interference Mitigation through Power Control in High Density 802.11 WLANs. In *IEEE INFOCOM*, 2007.
11. T. Razafindralambo, I. G. Lassous, L. Iannone, and S. Fdida. Dynamic Packet Aggregation to Solve Performance Anomaly in 802.11 Wireless Networks. In *ACM MSWiM*, October 2006.
12. ANSI/IEEE 802.11-Standard. 1999 edition.
13. H. Kim, S. Yun, I. Kang, and S. Bahk. Resolving 802.11 Performance Anomalies through QoS Differentiation. In *IEEE Comm. Letters, Vol. 9, No. 7*, July 2005.
14. P. Bellavista, A. Corradi, and L. Foschini. The MUM Middleware to Counteract IEEE 802.11 Performance Anomaly in Context-aware Multimedia Provisioning. In *International Journal of Multimedia and Ubiquitous Engineering, Vol. 2, No. 2*, July 2007.
15. Hierarchical Token Bucket. http://luxik.cdi.cz/∼devik/qos/htb/.
16. A. Vlavianos, E. Law, I. Broustis, S. V. Krishnamurthy, and M. Faloutsos. Assessing Link Quality in IEEE 802.11 Wireless Networks: Which is the Right Metric? In *IEEE PIMRC*, 2008.
17. J. Dunn, M. Neufeld, A. Sheth, D. Grunwald, and J. Bennett. A Practical Cross-Layer Mechanism For Fairness in 802.11 Networks. In *IEEE BROADNETS*, 2004.
18. M. Portoles, Z. Zhong, and S. Choi. IEEE 802.11 Downlink Traffic Shaping Scheme for Multi-User Service. In *IEEE PIMRC*, 2003.
19. L. Iannone and S. Fdida. Sdt. 11b: Un Schema a Division de Temps Pour Eviter l'anomalie de la Couche MAC 802.11b. In *CFIP*, April 2005.
20. S. Yoo, J. Choi, J.-H. Hwang, and C. Yoo. Eliminating the Performance Anomaly of 802.11b. In *ICN*, 2005.
21. D. Yang et al. Performance Enhancement of Multi-Rate IEEE 802.11 WLANs with Geographically-Scattered Stations. In *IEEE Trans. Mob. Comp., Vol. 5, Iss. 7*, July 2006.
22. G. Tan and J. Guttag. Time-based Fairness Improves Performance in Multi-rate WLANs. In *ACM USENIX*, 2004.
23. L. Guo, X. Ding, H. Wang, Q. Li, S. Chen, and X. Zhang. Exploiting Idle Communication Power to Improve Wireless Network Performance and Energy Efficiency. In *IEEE INFOCOM*, 2006.
24. V. Bahl, R. Chandra, P. P. C. Lee, V. Misra, J. Padhye, D. Rubenstein, and Y. Yu. Opportunistic Use of Client Repeaters to Improve Performance of WLANs. In *ACM CONEXT*, 2008.
25. W. Xu, W. Trappe, Y. Zhang, and T. Wood. The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In *ACM MOBIHOC*, 2005.
26. R. Gummadi et al. Understanding and Mitigating the Impact of RF Interference on 802.11 Networks. In *ACM SIGCOMM*, 2007.
27. V. Navda et al. Using Channel Hopping to Increase 802.11 Resilience to Jamming Attacks. In *IEEE INFOCOM mini-conference*, 2007.
28. W. Hu, T. Wood, W. Trappe, and Y. Zhang. Channel Surfing and Spatial Retreats: Defenses Against Wireless Denial of Service. In *WISE*, 2004.
29. ISM Wide-band Jammers. http://69.6.206.229/e-commerce-solutions-catalog1.0.4.html.
30. ISA: Users fear wireless networks for control. http://lists.jammed.com/ISN/2007/05/0122.html.
31. K. Pelechrinis, C. Koufogiannakis, and S.V. Krisnamurthy. Gaming the Jammer: Is Frequency Hopping Effective? In *WiOpt*, June 2009.

32. R. Vedantham, S. Kakumanu, S. Lakshmanan, and R. Sivakumar. Component Based Channel Assignment in Single Radio, Multi-channel Ad Hoc Networks. In *ACM MOBICOM*, 2006.
33. S. Kandula, K.C-J. Lin, T. Badirkhanli, and D. Katabi. FatVAP: Aggregating AP Backhaul Capacity to Maximize Throughput. In *USENIX NSDI*, 2008.
34. W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming Sensor Networks: Attacks and Defense Strategies. In *IEEE Network*, May/June 2006.
35. G. Lin and G. Noubir. On Link Layer Denial of Service in Data Wireless LANs. In *Wireless Communications and Mobile Computing*, May 2003.
36. G. Noubir and G. Lin. Low-power DoS Attacks in Data Wireless LANs and Countermeasures. In *ACM MOBIHOC (poster)*, 2003.
37. W. Xu, W. Trappe, and Y. Zhang. Anti-jamming Timing Channels for Wireless Networks. In *ACM WiSec*, 2008.
38. G. Noubir. On Connectivity in Ad Hoc Network under Jamming Using Directional Antennas and Mobility. In *Wired/Wireless Internet Communications, Vol. 2957/2004, pp. 186-200, 2004*.
39. M. Li, I. Koutsopoulos, and R. Poovendran. Optimal Jamming Attacks and Network Defense Policies in Wireless Sensor Networks. In *IEEE INFOCOM*, 2007.
40. M. Cagalj, S. Capkun, and J.-P. Hubaux. Wormhole-based anti-jamming techniques in sensor networks. In *IEEE Trans. on Mobile Computing, vol. 6, no. 1, pp. 100-114*, Jan. 2007.
41. Y. W. Law, L. van Hoesel, J. Doumen, P. Hartel, and P. Havinga. Energy Efficient Link-Layer Jamming Attacks Against Wireless Sensor Network MAC Protocols. In *Proc. ACM Security Sensor Ad-hoc Networks (SASN)*, 2005.
42. R. Mallik, R. Scholtz, and G. Papavassilopoulos. Analysis of an On-Off Jamming Situation as a Dynamic Game. In *IEEE Trans. Commun., vol. 48, no. 8, pp. 1360-1373*, Aug. 2000.
43. B. Awerbuch et al. A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. In *PODC*, 2008.
44. A.D. Wood, J.A. Stankovic, , and G. Zhou. DEEJAM: Defeating Energy-Efficient Jamming in IEEE 802.15.4-Based Wireless Networks. In *IEEE SECON*, 2007.
45. M. Acharya, T. Shamra, D. Thuente, and D. Sizemore. Intelligent Jamming Attacks in 802.11b Wireless Networks. In *OPNETWORK conference*, 2005.
46. D. Thuente and M. Acharya. Intelligent Jamming in Wireless Networks with Applications to 802.11b And Other Networks. In *MILCOM*, 2006.
47. Rude/Crude measurement tool. http://rude.sourceforge.net/.
48. Soekris-net4826. http://www.soekris.com/net4826.htm.
49. A. Jardosh et al. IQU: Practical Queue-Based User Association. In *ACM MOBICOM*, 2006.
50. A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-Management in Chaotic Wireless Deployments. In *ACM MOBICOM*, 2005.