

Understanding and Mitigating the Impact of RF Interference on 802.11 Networks

Ramakrishna Gummadi*

David Wetherall†‡

Ben Greenstein†

Srinivasan Seshan§

*University of Southern California

† Intel Research Seattle

‡University of Washington

§CMU

ABSTRACT

We study the impact of RF interference on 802.11 networks, both from devices such as Zigbee and cordless phones that increasingly crowd the 2.4GHz ISM band, and from devices such as wireless camera jammers and cheap 802.11 devices that seek to disrupt 802.11 operation. We find that commodity 802.11 equipment is surprisingly vulnerable to certain patterns of weak or narrow-band interference. This enables us to disrupt a link with an interfering signal whose power is 1000 times weaker than the victim’s 802.11 signals, or to shut down a multiple AP, multiple channel managed network at a location with a single radio interferer. We find the underlying causes of these vulnerabilities range from MAC layer driver implementation strategies to PHY layer radio frequency implementation strategies. Because they cannot be overcome by simply changing 802.11 operational parameters, we explore rapid channel hopping as a strategy to withstand RF interference. We prototype a channel hopping design using PRISM NICs and show experimentally that it greatly improves the ability to tolerate RF interference at a reasonable cost in terms of switching overheads.

1. INTRODUCTION

Our reliance on wireless communications such as 802.11 is increasing. Wireless technology is now used as an alternative to wired networks in enterprises [17], to enable mobility in safety critical settings like hospitals, and to provide city-wide Internet access [10]. In each of these cases, high network availability is desirable. Unfortunately, by their nature, wireless transmissions are vulnerable to RF (Radio Frequency) interference from various sources. This weakness is a growing problem for technologies that operate in the ISM frequency bands, as these bands are becoming more crowded over time [14]. 802.11b/g networks which use the 2.4GHz ISM band now compete with a wide range of wireless devices that includes 2.4GHz cordless phones, Bluetooth headsets, Zigbee (IEEE 802.15.4) embedded devices, 2.4GHz RFID tags, proprietary transmitters such as the ANT radios [2] and Cypress “WirelessUSB” peripherals [6].

Although the unlicensed ISM bands require no coordination between the deployers of devices, they do not permit all forms of behavior. To promote co-existence, devices that use

the ISM band must meet a number of FCC and ITU regulations that limit transmission power and force nodes to spread their signals. Wireless technologies often have mechanisms in their MAC and PHY that go beyond the basic FCC/ITU rules to improve co-existence. For example, 802.11 uses carrier sense to detect and defer to 802.11 and other transmitters. Similarly, Bluetooth adaptively hops frequencies to decrease interference on 802.11 [1]. However, ISM band co-existence and additional precautions have not prevented a range of interference problems. First, we can not expect to have designs that address the n^2 combinations of wireless technology interactions. In fact, there are reports of interference between devices that are specifically designed to co-exist (e.g., 802.11 and Bluetooth [5]). Unfortunately, more generic mechanisms for politely accommodating other transmitters, such as carrier sense in 802.11, also increase susceptibility to interference from other technologies.

Our goal is to explore the impact of interference on 802.11 links, along with techniques to make 802.11 more resistant to interference such that its performance degrades gracefully as the interference increases. We would like 802.11 nodes to be robust to devices that co-exist with them in the ISM band. We call this class of devices *selfish* interferers since they run their own protocol for their own benefit. It is also important that 802.11 nodes tolerate devices that seek primarily to deny service and not perform any useful work of their own. Such devices include wireless jammers [9] and commodity 802.11 devices that can be easily subverted to generate interference. We call this class of devices *malicious* interferers.

To study such interference, we subject an 802.11 network consisting of a single AP and a single nearby client to interference using commonly available RF sources and measure the effect on client/AP performance. 802.11 already uses many mechanisms to mitigate noise and interference, so it is natural to ask whether 802.11 links are already as robust to interference as can reasonably be expected. These mechanisms include: 1) a MAC protocol that avoids collisions; 2) lower transmission rates that accommodate lower signal-to-interference-plus-noise (SINR) ratios; 3) signal spreading that tolerates narrow-band fading and interference; and 4) PHY layer coding for error correction. However, we are aware of little published work that we can use to answer this

question. Past studies [20, 22, 24, 25] have considered co-channel and adjacent channel interference, in which the RF sources follow 802.11 physical and MAC layer standards. We also consider interferers that do not follow 802.11 standards, though they do fall under broad FCC regulations.

Our experimental results confirm anecdotal evidence that a range of selfish and malicious interferers (802.11 waveforms, Zigbee, a wireless camera jammer, a cordless phone) cause 802.11 performance to degrade significantly. Surprisingly, we found that even highly attenuated signals from malicious devices can cause severe losses at the receiver. We identify a number of properties of a typical NIC's (Network Interface Card's) implementation of the 802.11 PHY and MAC layers that is to blame for this poor performance. This leads us to extend the SINR model of successful packet transmissions to account for these effects. We validate this model by using experimental data to set parameters, and by checking that the model predicts the experimental outcomes. In particular, the model shows why some likely interference mitigation techniques such as high SINR do not help. It also explains why shifting to adjacent though non-orthogonal channels does help.

Our experiments and model show that obvious measures that might be expected to mitigate interference are not helpful due to reception path limitations. For example, high sender transmit power, large channel bandwidth compared to a narrow-band interferer, high receiver selectivity, and multi-antenna and spatial diversity techniques used in the new 802.11n do not gracefully tolerate interference. However, we do find that existing 802.11 implementations are able to tolerate interference when it is modestly off the center of the frequency channel that they are using.

Based on these observations, we design a channel hopping scheme and evaluate its ability to withstand interference. We use commodity (PRISM) chipsets to prototype our design. In it, clients and the AP switch to a pseudo-random channel rapidly ($250\mu\text{s}$ channel switching latency), and occupy it for a short period (10ms dwell period) before switching again. This makes it difficult for both selfish and malicious devices to jam a link for an extended period. This is because they must first find the channel that the link is using at a given time (or jam all channels, which we show is considerably more expensive). We find the overhead of channel hopping to be acceptably small, and the improvement in performance under interference to be large. Without hopping, the effect of a single interferer is catastrophic. With hopping, even three interferers jamming all three orthogonal channels cannot appreciably degrade performance. Our results suggest that channel hopping could be incorporated into 802.11 networks to make them more robust to otherwise harmful interference, with little impact when there is no interference.

In this paper, we make three contributions. First, we quantify the extent and magnitude of 802.11's vulnerability to interference, and relate the causes of such vulnerability to design limitations in commodity NICs. Second, we extend

the SINR model to capture these limitations, and quantify how our extended version can be used to predict the high interference degradation with even weak and narrow-band interferers seen in practice. We also use the model to show that changing 802.11 operational parameters would be ineffective at mitigating this degradation, while channel hopping can be helpful. Third, we implement and evaluate a rapid channel hopping scheme that can withstand even multiple strong interferers in a realistic setting, at a reasonable cost in terms of channel switching overheads.

The rest of the paper is organized as follows. We briefly review 802.11 in the next section, then describe our experimental setup in Section 3. We describe our experiments to gauge the effects of interference in Section 4, and extend the SINR model to capture these effects in Section 5. Our channel hopping solution is developed in Section 6. We then consider related work in Section 7 and conclude in Section 8.

2. 802.11 BACKGROUND

We briefly review 802.11 as it is relevant to our work.

802.11 nodes follow a contention-based CSMA/CA MAC defined by the IEEE standard. Normally, the 802.11 radio is in receive mode. When a node has a packet to send, it enters the transmit mode and waits for a certain time period to make sure the medium is free (CSMA). It uses a Clear Channel Assessment (CCA) module that may be configured in several modes to make this determination. In Mode 1, the transmitter declares the medium busy if it detects any signal energy above the Energy Detect (ED) threshold. In mode 2, it declares a busy medium if it detects any valid 802.11-modulated signal. In mode 3, a busy medium is declared only when a valid 802.11-modulated signal that exceeds the ED threshold is detected. Normally, mode 2 is used.

If the CCA module declares the medium to be free, the packet is sent. If it is busy, the transmitter defers the transmission for a random number of $20\mu\text{s}$ slots selected between 1 and the Contention Window (CW), and repeats the CCA procedure. The CW is doubled with successive deferrals, up to a maximum of 127 slots; the packet is sent if this maximum is reached regardless of whether the medium is busy. The CW is reset to a minimum value after a transmission.

Receivers send an ACK packet within a fixed time limit to acknowledge the receipt of a non-broadcast data packet that passes the CRC check for data integrity. If the transmitter does not receive an ACK, it considers the packet (or its ACK) lost. It then retransmits the packet by re-inserting it at the front of the transmission queue and treating it as a new packet. Retransmission can be repeated up to seven times, after which the packet is dropped. Optionally, nodes can precede data packets with a RTS/CTS exchange to reduce the likelihood of interference by hidden terminals, but most implementations choose not to do so in practice because the costs outweigh the benefits.

The 802.11 MAC also defines management packets, the most relevant here being beacons and probes. An AP period-

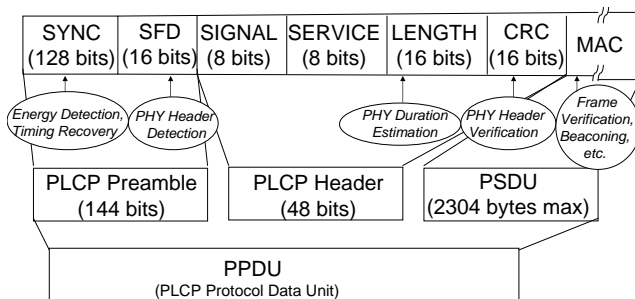


Figure 1—802.11 PHY encapsulation and its usage at the receiver.

ically (~ 100 ms) broadcasts beacons to assist clients with association, roaming, synchronization, power-saving and other tasks. Beacons carry an 8-octet timestamp field so that the client’s NIC can synchronize its clock with the AP to meet the timing constraints of the 802.11 MAC. Probe packets are sent by a client to discover APs.

Reception at a node can be explained in terms of the PLCP (Physical Layer Convergence Protocol) headers that encapsulate packets (shown in Figure 1). Processing steps are shown as ovals. To begin, a preamble of a *SYNC* bit-pattern triggers the energy detection circuitry that alerts the receiver to an incoming transmission. This bit-pattern is also used to extract symbol timing. It is always transmitted at 1Mbps. 802.11b/g uses either a long preamble that transmits the PLCP header (Figure 1) at 1Mbps or a short preamble that transmits the PLCP header at 2Mbps, regardless of the transmit speed of the MAC frame itself. A long preamble is shown in the figure. The Start Frame Delimiter (*SFD*) is a specific 16-bit pattern (0x07cf with long preambles) that signifies the start of PLCP data. In the PLCP header, the *LENGTH* field contains the packet length, which is used with bit rate information in the *SERVICE* field to determine the overall duration of the packet. To complete the PLCP processing, the receiver computes a CRC over the header. It generates a physical-layer error if the header is corrupted. The MAC frame follows and it includes a separate CRC over the MAC contents. The receiver generates a separate MAC-layer error if the MAC is corrupted. In Section 4, we study how interference can disrupt the processing of these PHY and MAC functions.

An 802.11b/g transmission occurs on one of 11 overlapping channels in the 2.4GHz North American ISM band; the band is wide enough for three orthogonal channels. On a given channel, 802.11 offers a large choice of rates and modulations that trade off performance for interference tolerance. 802.11b rates are 1Mbps (Differential Binary Phase Shift Keying, DBPSK), 2Mbps (Differential Quadrature Phase Shift Keying, DQPSK), 5.5Mbps (Complementary Code Keying, CCK), and 11Mbps (CCK). The 1 and 2Mbps rates use Direct Sequence Spread Spectrum (DSSS) to spread their signals across the entire 22MHz channel bandwidth and increase noise immunity. The spreading sequence, the 11-bit Barker code, has low auto-correlation to tolerate multipath

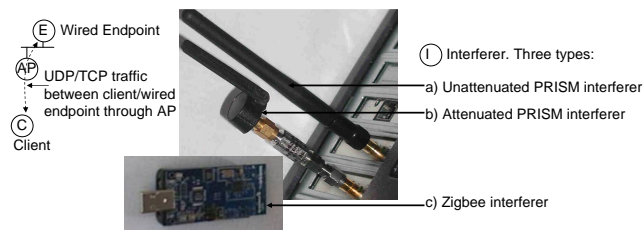


Figure 2—Experimental setup with three interferers.

conditions, and gives a processing gain of 10.4dB. CCK in the 5.5 and 11Mbps rates handles both modulation and spreading. 802.11g, like 802.11a, uses Orthogonal Frequency Division Multiplexing (OFDM) for modulation. 52 tightly-spaced (0.3125MHz apart) orthogonal sub-carriers, of which 48 are for data, carry data at various rates ranging from 54Mbps down to 6Mbps depending on channel conditions.

3. EXPERIMENTAL SETUP

For our experiments, we use a simple network setup (Figure 2) that consists of an AP, client, and selfish or malicious interferer. This is to clearly expose low-level interference effects. We ran experiments with PRISM, Atheros and Intel NICs as described below to ensure that we do not focus on implementation deficiencies that are easily corrected.

Client and AP. The client is a Linux laptop equipped with 802.11 NICs from PRISM (802.11b), as well as Atheros and Intel (802.11a/b/g), in PCMCIA and mini-PCI formats. The AP is a Linux laptop with either a PRISM 2.5 in 802.11b mode (using the *HostAP* driver) or an Atheros AR5006X in 802.11b/g mode (using the *MadWifi* driver). The majority of current 802.11 NICs belong to one of these three architectures, and all implement the 802.11 PHY in hardware, and at least the time-critical parts of the 802.11 MAC in firmware.

During our early experiments, we found that the NIC is highly sensitive to beacon losses at the client. During beacon loss periods, a NIC rapidly begins looking for other APs to associate with and is prone to lock-ups under high loss. We mask these effects to observe other interference effects by disabling beacon transmission at the AP and manually assigning the MAC address of the AP on the client. This has the side-effect of turning off NIC clock synchronization, but initial synchronization to the AP’s clock comes from a probe reply message, and we did not see subsequent adverse effects. Note that the channel hopping technique that we propose in Section 6 employs beaconing.

Interferers. We use four qualitatively different sources of interference in our experiments (Table 1). We use two malicious devices (PRISM-based interferer and video camera jammer) and two selfish devices (a Zigbee sensor node and a cordless phone). To understand degradation effects (Section 4), we use the cheap and ubiquitous PRISM 802.11 NICs with custom software, and Zigbee nodes as interferers. Our Zigbee nodes are sensor motes equipped with Chip-

Interferer	Power(dBm)	BW(MHz)	Range(m)
PRISM 2.5	[-20, 20]	22	~30
Video camera jammer	30	1, FH	~20
CC2420 (Zigbee)	[-24, 0]	5	~6
Cordless phone	20	0.003, FH	~2

Table 1—Interferers and their characteristics.

con CC2420 radios [8], which implement the Zigbee-PHY in hardware and the Zigbee-MAC in software. To evaluate our mitigation strategies (Section 6), we also use a wireless video camera jammer [9] and a cordless phone.

In Table 1, the power column gives the power output by an interferer’s radio before antenna gain. To control the transmit power of the PRISM interferer over a wide range (40dB, or a factor of 10,000), we use hardware attenuators. For Zigbee, we change the power levels in software. In the BW (Bandwidth) column, *FH* means the device frequency hops the entire 2.4–2.4835GHz band. The range column in Table 1 shows the approximate range we found the interferer to be highly effective (*i.e.*, halted TCP transfers between the wired endpoint and the client in Figure 2).

The PRISM interferer is a Linux desktop with PRISM PCI NICs, as shown in Figure 2, with custom software. We chose it because the PRISM firmware provides a low-level interface that can generate arbitrary 802.11-modulated continuous 16-bit patterns as the MAC data. Such RF patterns are valid modulated 802.11 signals, but not valid 802.11 PLCP or MAC units. We use a user-level program to generate and count the duration of these interference patterns; they cannot be measured externally using packet sniffers because sniffers typically only decode frames with valid MAC data.

The Zigbee interferer outputs 128-byte packets, without any transmission control. The wireless camera jammer is a commercially available device that uses frequency hopping to block all 802.11b/g channels. The cordless phone is a Panasonic brand commodity device.

For our experiments in Section 4, we place the interferer to ensure that its signals at the AP and client are more attenuated than the AP to client signals at all times. We verified this by measuring the signal and noise (which includes interference) powers at the AP and client. This is to avoid overstating the effects of interference. The client and the AP are physically closer to each other than the interferer, and have a direct line-of-sight to each other to mitigate small-scale path loss considerations such as multipath and fading. However, to evaluate our channel hopping design in Section 6, we use a more realistic setup with multiple, non-line-of-sight clients and multiple interferers whose signals can be stronger than the AP and client.

Tests and Metrics. The tests were conducted in a lab that is part of a 30m×30m office floor. There were other 802.11 networks nearby, but we ran our experiments mostly when there was little external traffic. Each test consists of the client doing a one-way UDP or a TCP transfer of several megabytes

between itself and a wired source or sink E through the AP, as shown in Figure 2. The packet size is 1500 bytes, and we provisioned enough socket buffers at the end hosts and enough forwarding buffer at the AP so that there were no packet losses inside the nodes themselves.

We measure overall performance in terms of throughput and latency. For each test, we measure kernel-level end-to-end packet transmissions and receptions at one-second intervals. To investigate performance effects, we also collect many 802.11 statistics at the AP and the client.

4. CAUSES AND EFFECTS OF INTERFERENCE

This section presents the results of our interference experiments, and shows how design choices made in commodity NICs explain our results. We categorize our results into three main classes: limitations related to timing recovery, limitations related to dynamic range selection, and limitations related to PLCP header processing. Each of these limitations leads to high packet loss and, consequently, low throughput.

We test with NICs from different vendors (PRISM, Atheros and Intel depending on the test) to check that these effects are not implementation artifacts. We report results mainly for the PRISM NIC due to space limitations. We also test with 802.11g and 802.11n to check that these effects are not 802.11b PHY artifacts that can be overcome with modulation schemes that have different receiver parameters for the processing chain. Finally, we show that if the interferer is modestly away from the center frequency of an 802.11 channel even though it is within the receive band (*e.g.*, separated by 5MHz or more, as are two adjacent 802.11 channels), then the interference is significantly less damaging.

4.1 Timing Recovery Interference

Sender clock extraction is done in the Timing Recovery module in Figure 3. If this module fails to lock onto the sender’s clock, the receiver will sense energy, but not recognize it as valid modulated *SYNC* bits. Synchronization begins when the receiver detects the *SYNC* bit pattern of 128 scrambled 1s (long-preamble) or 56 scrambled 0s (short-preamble). They are always sent at 1Mbps regardless of the data rate for the rest of the packet. As shown in Figure 3, the transmitter scrambles the PLCP preamble that includes the *SYNC* bits to remove DC-bias (in the Scrambler module in Figure 3), modulates them using DBPSK modulation (in the Modulator module), and spreads them (in the Barker Spreader module). At the receiver, the RF signal is digitized into 6-bit samples (in the Analog-to-Digital Converter or ADC module), and these samples are processed to recover the sender’s clock so that the rest of the receiver components can work on aligned signal samples.

We consider the impact of a PRISM interferer that emits a continuous all 1s pattern, which directly interferes with the receiver’s Timing Recovery module. This pattern is scrambled, modulated, and spread by the interferer’s NIC in the

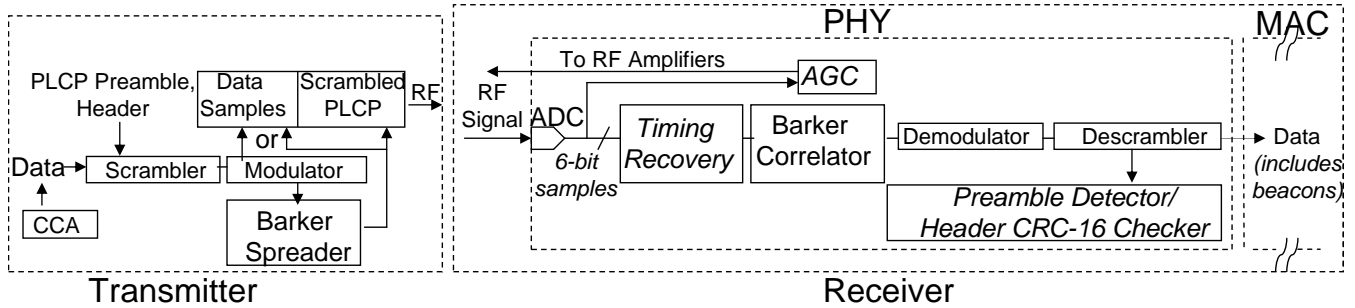


Figure 3—The PHY processing chains at the transmitter and the receiver. The components in the receiver vulnerable to interference are shown in italics.

same way that the transmitter’s *SYNC* bits are. Since the interferer’s clock and the transmitter’s clock are unsynchronized, the Timing Recovery module at the receiver cannot lock onto the transmitter’s clock. The receiver therefore only records energy detection events, but does not detect any packet transmissions. We also experimented with an all 0s pattern to interfere with devices that use short preambles, with substantially similar results.

We plot the throughput and latency for UDP traffic under this interference pattern in Figure 4. This graph is for a single client to AP flow and PRISM NICs; Atheros results are qualitatively similar. Throughput is on a log-scale. It includes confidence intervals, but they are generally too small to see as they are all within 5% of the actual values. Latency is measured as the time that the transmitter handles each packet, before sending or dropping it. ()

The graph shows that, as the interference increases beyond 12dBm (or 16mW¹), the receiver fails to lock onto any packet, and the throughput drops to zero. For comparison, the AP and client transmissions are at 20dBm between physically closer devices, and so are expected to be significantly higher than this level of interference. Link latency also increases with loss, as expected. Surprisingly, the plot shows that even small amounts of interference cause significant loss (e.g., -20dBm, or 0.01mW interferer power reduces throughput by a factor of four) even though the SINR is high. We investigate the reasons for this sensitivity to attenuated interference in the next subsection.

We also found that higher layer effects can exacerbate lower layer ones. Specifically, clients disconnect from their APs under moderate packet loss. This is because the MAC firmware of NICs like PRISM and Intel is especially sensitive to three or more consecutive beacon losses. This allowed us to use a single radio interferer to disconnect clients associated with different APs operating on multiple channels. We simply cycled through all 11 channels and emitted interference briefly on each channel. In one experiment, we

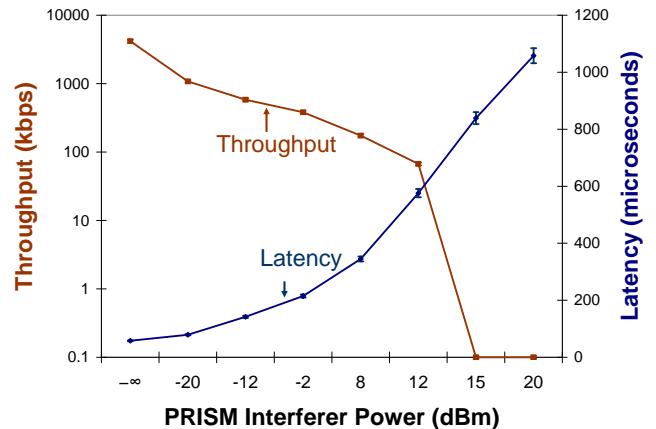


Figure 4—Throughput and latency vs. interferer power caused by interference affecting timing recovery.

made the PRISM interferer switch channels rapidly using a low-level PHY interface, while continuously emitting the all 1s interference pattern. There were six APs belonging to a single managed network, each listening on a different channel. The APs were spread around an office floor that was 30m long and 30m wide. Interference caused all clients in the office, who were connected to different APs, to disconnect from their APs within a short period (less than 5s), and remain disconnected. This is because they could not reliably receive beacons from any AP, even though a client was within transmission range of several APs on average.

4.2 Dynamic Range Selection Limitation

Receivers need to decode packets over a very large range of signal strengths: the strongest signals are typically around -10dBm, while weak signals can be -70dBm or less, a range of 60dB, or a factor of 10⁶. To work over this range, the receiver normalizes these signals internally into a fixed range. The fixed range is designed so that, after taking the average background noise into account, the Analog-to-Digital Converter (the ADC module in Figure 3) can make the best use of the fixed-width bits that are available to represent the

¹The relationship between dBm and milliwatts is:
 $P = 10^{(x/10)} \text{mW}$, where P is power in mW and x is in dBm.

digital samples of the signal. In PRISMs, these samples are 6-bit wide, representing 64 different voltage levels [3]. An automatic gain control unit (the AGC module in Figure 3) samples these voltage levels during the PLCP preamble processing, and controls the gain of the RF amplifiers so that the signal samples can occupy the entire ADC range.

For cost and complexity reasons, there are two limitations of such a design in commodity NICs such as PRISM and Intel that we find lead to significant interference effects:

- The AGC is fairly simple in practice, checking to see if the signal voltage level, during the time it is sampling the *SYNC* bits, is greater than a certain voltage threshold. If so, the signal is considered strong, and the AGC asks the RF amplifiers to subtract a 30dB gain from the incoming signals. However, the RF amplifiers do not proportionately attenuate interference, e.g., with filters. Instead, they use linear voltage comparators to linearly subtract signal voltages. This removes the high-order bits of the signal voltage, which attenuates the signal but does not attenuate interference carried in the low-order bits of the signal voltage. ()
- Gain control and dynamic range selection are only done once per packet, during the PLCP preamble processing (*i.e.*, just before the PLCP header is about to be processed). This means that if interference is introduced after the gain control is done, the 6-bit signal voltage levels that are output at the ADC are not adjusted to cope with this interference, and can overflow. Similarly, if interference is removed after gain control, these voltage levels of the signal can underflow.

This range selection process can be undermined by both attenuated and narrow-band interferers. We consider an interference pattern consisting of a random 16-bit pattern, which is turned on for a short period (5ms), and then switched off for another short period (1ms). This process is then repeated with another random pattern. These random patterns interfere with the dynamic range selection because the receiver can not calibrate the signal power or the noise floor correctly with such rapid on-off patterns. This means when such an interference is added to a strong signal that has been attenuated, it can cause the signal samples at the output of the ADC to overflow, as described above. Similarly, when the random interference pattern is removed from a strong signal that has been attenuated, it can cause underflow of signal samples, because the receiver had estimated a high noise floor while decoding the preamble that had this interference added to it. Thus, the net effect of such interference patterns is to cause CRC errors either in the PLCP header or, if the PLCP header is received correctly, in the data payload. This leads to packet corruption, which, in turn, leads to high packet loss. ()

These random patterns can be emitted by both malicious interferers, such as our PRISM-based jammer, and selfish interferers, such as Zigbee nodes that transmit sensor data in rapid bursts. While these patterns can also cause some

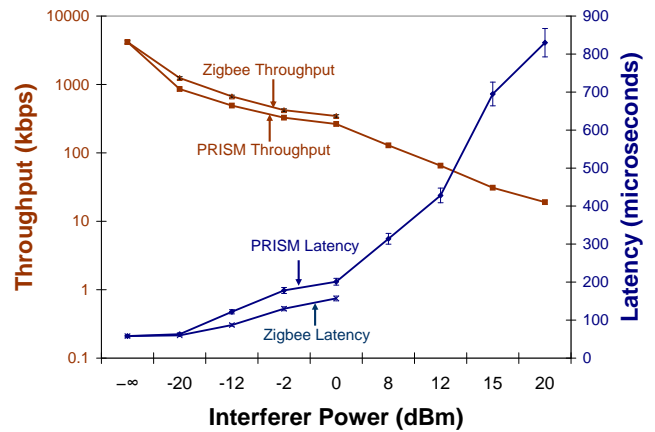


Figure 5—Throughput and latency vs. interferer power caused by interference affecting dynamic range selection.

timing recovery failures, as with a continuous all 1s pattern, their main effect is to cause packet losses under weaker interference conditions.

We show the performance impact in Figure 5 for the same setup as previously. We plot the throughput (on a log-scale) and latency for two interferers that output random patterns in bursts. The output range of the Zigbee radio is restricted to $[-24, 0]$ dBm. It can be seen from the throughput graphs that even a small amount of interference is effective at causing heavy losses. Further, such interference patterns are effective with both selfish and malicious interferers, because such interference artificially lowers the working SINR rather than relying on any properties that are specific to 802.11. For this same reason, the PRISM interferer does not cause the link throughput to drop to zero at power levels above 12dBm, unlike timing recovery interference (Figure 4). However, it causes slightly lower link throughput at power levels 12dBm and below because the on-off interference patterns can affect packet processing during the entire transmission period of a packet, whereas timing recovery interference is only effective during *SYNC* processing. Link latency increases with interferer power and is slightly higher with the PRISM interferer than with the Zigbee interferer. This is because PRISM also induces CCA backoffs in Mode 2 (the default mode in most NICs) because it outputs modulated 802.11b energy.

While the link fares marginally better under Zigbee interference than under PRISM interference, we were surprised to find that a non-802.11 narrow-band interferer could be so effective in practice, especially because Zigbee channels are slightly (2MHz or more) offset from 802.11 channels. We found that the cause to be the non-linearity in receiver sensitivity. The sensitivity of the receiver’s RF amplifiers (there are two or more of them in most NICs) drops off non-linearly as the frequency separation between the interferer and the center frequency of the 802.11 channel the amplifiers are tuned to increase. This drop-off is small near the center frequencies (for example, at 2MHz, the interference attenua-

tion is around 10dB in the PRISM receivers), but increases non-linearly as the frequency separation increases (the interference attenuation increases to around 30dB at 5MHz in the PRISM receivers). This weights signal energy close to the center frequency disproportionately higher than energy in the receive band but away from the center.

4.3 Header Processing Interference

We also discovered that we could cause loss by interfering with the mechanism that starts packet processing at the receiver. To do this, we continuously transmit the modulated 16-bit data value used by the *Start Frame Delimiter (SFD)* field (Figure 1) in the PLCP preamble. This field signals to the receiver that the PLCP header is about to be sent. The receiver is expected to have initialized its processing chain (*i.e.*, ensured that the AGC, the Barker Correlator, the Demodulator and the Descrambler modules are ready) by this time. The *SYNC* bits are designed to allow receivers sufficient time to do so. This means that, in practice, receivers are ready for the *SFD* pattern before it arrives. If the receiver’s Preamble Detector module in Figure 3 sees the *SFD* pattern from the interferer before it sees it from the transmitter, it starts processing the header before the actual header from the transmitter arrives at the receiver. This means that it assembles the header fields such as *LENGTH* and *CRC* (Figure 1) from the wrong samples. Consequently, the CRC that the Header CRC-16 Checker module computes over such samples will not match what the receiver thinks is the *CRC* of the PLCP header. This results in the PHY header checksum error (a condition which is explicitly detectable on NICs based on the Atheros, PRISM, and Intel chips).

Surprisingly, this interference pattern works even when the interferer’s clock and the transmitter’s clock are not synchronized, and even when the transmitter is stronger than the interferer. This is because of the AGC gain limitations described in Section 4.2: the AGC module drops the transmitter’s signal by as much as 30dB, and the Timing Recovery module can therefore become synchronized to the interferer.

We plot the link throughput and latency under a PRISM interferer that generates continuous long-preamble *SFD* patterns in Figure 6 with the same setup as previously. Once again, the impact of interference is substantial for even attenuated interferers. The impact of this interference on throughput is more significant (by up to 40%) than either the timing recovery interference (Figure 5) or the dynamic range selection interference (Figure 6) for interferer power up to 8dBm. This is because the *SFD* interference pattern can affect throughput both by causing header CRC errors and, if that fails, by causing MAC CRC errors. Beyond 8dBm, the link throughput is higher with this pattern than the random interference pattern because the random pattern has a higher probability of disrupting a stronger signal than a fixed interference pattern has.

To interfere with devices that use short preambles, we also experimented with the short-preamble *SFD* pattern, with qual-

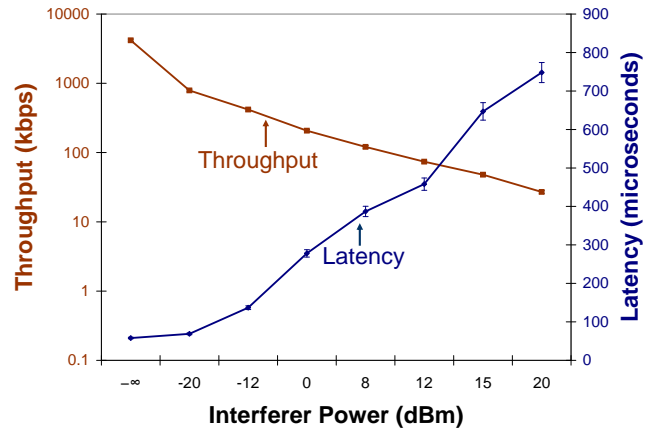


Figure 6—Throughput and latency vs. interferer power caused by interference affecting header processing.

itatively similar results.

4.4 Impact of Interference on 802.11g/n

While many of the components in the receiver path in Figure 3 are present in 802.11g and 802.11n, these new standards are different enough from 802.11b to question whether interference can decrease their link throughputs drastically as well. 802.11g does not use the Barker Correlator module, and the Demodulator module is quite different because it uses OFDM. Similarly, the new 802.11n standard applies spatial coding techniques, which use multiple transmitter and receiver antennas.

To tackle this question and establish the impact of interference, we subject transmissions from these new cards to the interference pattern used in Section 4.2. Recall, a PRISM interferer emitted a random data pattern in bursts, which prevented receivers from calibrating the signal power and the noise floor correctly. For 802.11g, we used Atheros NICs at the client and the AP in 802.11g-only mode, and for 802.11n, we used a D-Link DWA645 NIC and a D-Link DIR635 AP that implement the 802.11n draft standard.

In Figure 7, we plot the throughput and latency of UDP traffic sent over 802.11g and 802.11n links. Even though these links have high throughputs in the absence of interference, even small amounts of interference still cause substantial performance degradation. These new protocols share the same types of receiver limitations, such as limited dynamic range selection and non-linear receiver sensitivity.

4.5 Impact of Frequency Separation

We now examine the impact of interference as the interferer is progressively displaced from the center frequency of the transmitter and the receiver. We expect interference to be mitigated for two main reasons: the sensitivity of the RF amplifiers at the receiver falls off with frequency separation and the RF filters in the receiver remove interference power on frequencies that do not overlap the receiver’s frequencies.

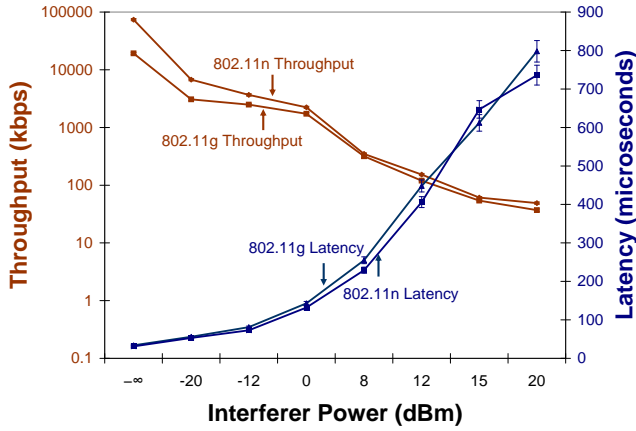


Figure 7—Throughput and latency vs. interferer power for 802.11g/n.

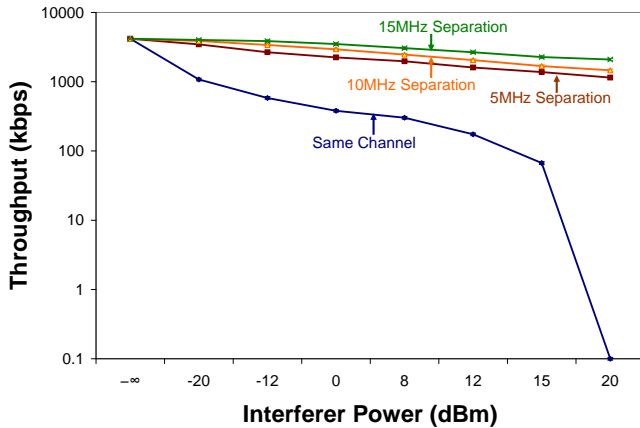


Figure 8—Throughput and latency vs. interferer power with frequency separation.

We move a PRISM interferer to adjacent 802.11 channels that overlap the client and AP transmissions (*i.e.*, these adjacent channels are not orthogonal). Figure 8 shows the impact of this frequency separation on link throughput. At 5MHz separation, the link throughput remains high (over 1Mbps) for all interferer output powers. At 10MHz separation, the link throughput is at least $\sim 33\%$ of the interference-free throughput, and at 15MHz separation, it is more than $\sim 50\%$. This tolerance to interference suggests that channel hopping may be an effective remedy in mitigating interference. We explore this idea in Section 6.

5. MODELING INTERFERENCE EFFECTS

This section presents a quantitative model for the interference effects we see, and uses it to explain why we see degraded performance even with attenuated and narrow-band interferers. Our model is an extension of the Signal to Interference plus Noise Ratio (SINR) model, and takes into account two important receiver limitations found in commod-

ity NIC designs, namely, dynamic range selection limitation due to the AGC, and receiver sensitivity non-linearity. As we pointed out in Section 4.2, these limitations allow weak and narrow-band interferers to be surprisingly effective.

The standard SINR model is widely used in simulators such as Qualnet and ns-2 to model the performance of wireless receivers. The basic idea is to compute the difference between the signal power and the combined power of interference and noise at the receiver. This SINR value is used to compute the bit-error rate, which is, in turn, used to calculate whether the receiver successfully receives a packet. The results of such simulations are reported to be in good agreement with real-world experiments [23]. But this simple SINR model does not predict the severe interference degradation that we see because it does not account for limitations of commodity NICs. For example, the SINR model predicts that packets will be received with high probability when the signal power at the receiver is at least 10dB greater than the interference power, yet we observe high loss.

To model these effects, we begin with the theoretical SINR model and extend it to include the limitations of real NICs that our experiments in Section 4 found to be significant. Using this extended model, we then predict the effects of changing 802.11 parameters such as bit rates, packet sizes, and modulation techniques. We experimentally confirm our predictions that such changes will not mitigate interference degradation, while moving to an adjacent channel will tolerate interference. ()

5.1 Extending the SINR Model

The standard SINR equation for each bit of a packet x that the receiver receives at time t is:

$$SINR(x, t) = \frac{S(x, t)}{I(x, t) + N_{env}} \quad (1)$$

Interference $I(\cdot)$ is sum of all undesirable signals $S(y, t)$ (both external interferers and self-interference due to multipath) that arrive at the receiver at time t :

$$I(x, t) = \sum_{y \neq x} S(y, t) \quad (2)$$

We can ignore multipath in our line-of-sight setup, so $I(\cdot)$ is simply the instantaneous interferer power.

The noise term in Equation 1 has several components, but is mainly the channel and antenna noise. It can be approximated as $N_{env} = kTB$, where k is the Boltzmann constant, T is the receiver temperature, and B is the signal bandwidth. At room temperature, for 22MHz 802.11b or 20MHz 802.11g, N_{env} is about -100dBm. For the 1Mbps rate (the slowest possible), we can then calculate using standard formulas that we need a signal-to-interference ratio of at least 10dB above this noise threshold of -100dBm in order to achieve a Bit Error Ratio (BER) of 10^{-6} (which roughly corresponds to a 1% packet loss with 1000-byte packets).

Accounting for processing gain. We need an SINR of at least 10dB to decode 802.11b signals correctly. Barker cod-

ing provides an addition 10.4dB processing gain, so that, theoretically, a signal can be -0.4dB weaker than an interferer, and still be received with only a 1% packet error rate. So far, we assume an ideal receiver and 1Mbps data rate, but this sets the lower bound on SINR.

Accounting for the AGC Behavior. As described in Section 4.2, the receiver’s Automatic Gain Control module can attenuate a strong signal by as much as 30dB in order to ensure that the signal power stays within the receiver’s processing range. It does this if the received signal power exceeds a threshold S_{\max} , a NIC-dependent constant. This is around -25dBm for the PRISM 2.5 NICs. This dynamic range limitation can thus lead to a loss of up to 30dB signal power at the demodulator. Thus, the signal power to the demodulator, $S_r(x, t)$, is actually:

$$S_r(x, t) = \begin{cases} S(x, t) - 30\text{dB}, & \text{for } S(x, t) > S_{\max} \\ S(x, t), & \text{for } S(x, t) \leq S_{\max} \end{cases} \quad (3)$$

Our model substitutes this equation into Equation 1. Since the SINR margin is -0.4dB with Barker coding, after this attenuation, the signal can not be demodulated unless the signal is now 29.6dB greater than the interferer. We will refer to this 29.6dB SINR requirement in the next section, where we apply this extended SINR model.

Accounting for Non-linearity in Receiver Sensitivity. As described in Section 4.2, the receiver’s amplifiers attenuate interference that is concentrated away from the center frequency of the selected 802.11 channel. However, this attenuation is not linear, and increases with the frequency separation between the receiver and the interferer. Thus, to accurately account for the impact of interference which is centered at a different frequency than the receiver, we need to integrate the interference power in Equation 2 with the receiver sensitivity over the entire frequency range $[f_1, f_2]$ that the receiver and the interferer overlap. Formally, $I(x, t)$ in Equation 2 is now:

$$I(x, t) = \sum_{y \neq x} \int_{f_1}^{f_2} R(f) S(y, t) df \quad (4)$$

where the receiver’s sensitivity at frequency f is $R(f)$.

We do not actually need to compute this weighted integral accurately, but can approximate it with the receiver sensitivity table from the data sheets of a particular receiver. For example, for PRISMs this sensitivity is about -10dB at 2MHz, and about -30dB at 5MHz, and This means that SINR effectively increases by 10dB if the interferer is displaced by 2MHz, and by 30dB if the displacement is 5MHz [13].

5.2 Applying the Model

We can use this model to explain the effects we found in Section 4 and to predict the effects of strategies that might be used to more gracefully tolerate interference. Specifically, we revisit the effects of an attenuated PRISM interferer, a normal (unattenuated) Zigbee interferer, and a normal PRISM

interferer on an adjacent-channel to build confidence in our model. We then predict and experimentally confirm the effect of varying 802.11 parameters such as packet sizes, rates and modulations, and coding gain. These are all plausible strategies for tolerating interference: small-size (100-byte) packets might be lost less often than normal-size (1500-byte) packets; low rates may be more robust than higher ones; and some modulation schemes such as BPSK, QPSK, and OFDM benefit from Forward Error Correction (FEC) coding to better withstand bit-errors in received packets. Unfortunately, none of these parameter changes are predicted or found to be effective! This leads us to the strategy of shifting frequencies that we explore as channel hopping in the next section.

As an aid to explain interference degradation seen in Section 4 and to make predictions about 802.11 parameters, we plot BER vs. SINR for all 802.11b modulations (Figure 9).

Attenuated PRISM. In one experiment, we measured a signal power of -18dBm and an attenuated PRISM interference (noise) power of -51dBm . Since the PRISM interferer also uses the same Barker code, it also incurs a processing gain. This means the SINR in this case is $-18 - (-51 + 10.4) = 22.6\text{dB}$, which is less than the required SINR of 29.6dB. This explains the heavy losses seen with even weak interferers. We will refer to this 7dB SINR shortfall with attenuated PRISMs below. \circ

Narrow-band Zigbee. Zigbee channels are separated from each other by 5MHz starting at 2.400GHz, and each channel occupies a 5MHz bandwidth. By design, the center frequencies of Zigbee and 802.11 are therefore always offset by at least 2MHz. The PRISM data sheet indicates that the receiver sensitivity at 2MHz offset is 10dB below center frequency [3]. We measured the Zigbee interference power at -35dBm . This gives us an SINR of $-18 - (-35) + 10 = 27\text{dB}$. Since this is below the required SINR of 29.6dB, the Zigbee narrow-band interferer also causes heavy losses in this case.

Adjacent-channel PRISM. An immediately adjacent 802.11 channel is 5MHz away from the center frequency of another 802.11 channel. This leads to three effects: the receiver sensitivity at 5MHz drops by more than 30dB; the interferer does not incur the Barker processing gain this time because the Barker correlator in the receiver does not correlate the interferer signal due to this 5MHz frequency offset; and some interferer power is filtered by the receiver filters. Concretely, we measured a noise power of -57dBm (after filtering) for the same attenuated PRISM. This means the SINR is now at least $-18 - (-57) + 30 = 69\text{dB}$, which is much larger than the required SINR of 29.6dB, and sufficient for even higher rate 802.11 modulations, even after relaxing the ideal receiver assumption (which typically incurs a 10dB penalty).

Changing Packet Sizes. We use the 7dB SINR requirement from the attenuated PRISM interferer example above. If we were to reduce packet size by a factor of 15 (from 1500 bytes to 100 bytes), we can see from Figure 9 that our SINR re-

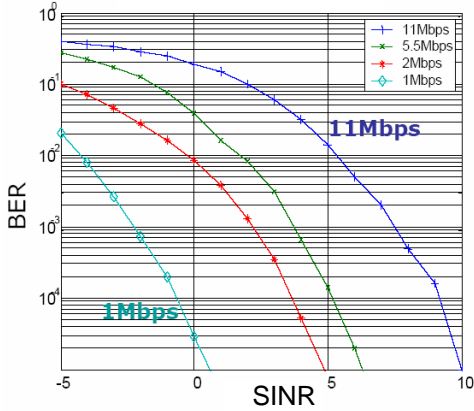


Figure 9—BER vs. SINR for 802.11b rates.

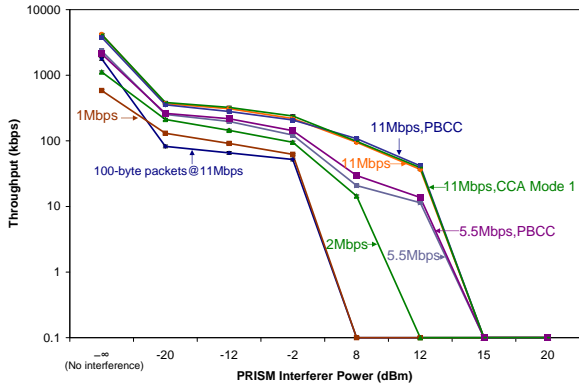


Figure 10—Throughput vs. interference with various packet sizes, rates, and modulations.

quirements drop by no more than 4dB for any modulation going from a BER of 10^{-5} to $\frac{10^{-5}}{15}$ (for example, the 1Mbps rate intersects the horizontal BER line of 10^{-5} at ~ 1 dB SINR, and the BER line of $\frac{10^{-5}}{15}$ at ~ -1.5 dB SINR, for an SINR drop of ~ 2.5 dB). Since we have an SINR shortfall of 7dB even with a 1Mbps modulation, we will still be short by $7 - 4 = 3$ dB. Thus, we can expect that changing packet sizes will not help much, as is indeed the case in practice (Figure 10). In this figure, we once again see that the link throughput decreases dramatically for all 802.11 parameters, including for 100-byte packets, when even small amounts of interference are introduced. Note that, in practice, the performance of UDP with small-size packets is worse than with large-size packets (the plot in the figure with the 11Mbps rate), because more packets induce extra CCA delays, without reducing packet losses significantly.

Changing Rates and Modulations. We consider whether changing the modulation schemes and rates may help. At the 1Mbps and 2Mbps rates, the UDP sender uses the DBPSK and DQPSK modulations, while the PRISM interferer uses DBPSK. This causes Barker gains for both. One question is whether rates that do not use Barker modulations, such as the 5.5Mbps and the 11Mbps CCK modulations, can im-

prove performance by not causing Barker gain for the interferer. To predict this situation, we look at Figure 9. If we use 5.5Mbps CCK, Figure 9 shows that we need an additional 7dB SINR over 1Mbps DBPSK (the 5.5Mbps and the 1Mbps rate curves intersect the horizontal BER line of 10^{-5} at around 7dB and 0dB respectively). Since our SINR shortfall with 1Mbps DBPSK is 7dB, we are still short by $7 + 7 - 10.4 = 4.4$ dB, and, so, CCK modulations should not help, as confirmed in practice in Figure 10.

Adding FEC. Finally, we consider whether FEC techniques such as convolution coding can help. PBCC (Packet Binary Convolution Coding) is one such coding that can be used with BPSK (5.5Mbps PBCC) or QPSK (11Mbps PBCC) modulation, and adds 4dB coding gain to these modulations [12]. It is supported in many NICs, such as Intel. In the attenuated PRISM example, we showed that the required SINR is 29.6dB, while the available SINR with BPSK modulation is only 22.6dB (it is lower with QPSK). Thus, even adding 4dB to BPSK modulation will not cover the 7dB SINR gap, and we can still expect high losses and low throughput, as we indeed confirmed in practice for both 5.5Mbps and 11Mbps PBCC rates in Figure 10.

Changing CCA Thresholds and Modes. It is apparent from our SINR model that changes to the CCA modes or thresholds will also not be effective. This is because they change behavior only at the transmitter, while we predict and observe losses at the receiver. Figure 10 confirms that changing the CCA mode to 1, with the CCA deferral threshold set high, has little effect on link throughput. Thus, the high CCA threshold decreases the number of deferrals per packet, without substantially affecting the link throughput. Additionally, we observed that altering CCA thresholds at only some clients caused unfairness in throughputs by up to 40%, while altering them at all clients essentially disabled the CSMA mechanism, leading to poor overall throughput.

6. RAPID CHANNEL HOPPING

In this section, we describe a rapid channel hopping (CH) technique. Our experiments show that channel hopping is the most effective way to tolerate interference gracefully without requiring hardware changes. Section 4 demonstrated that separating the interferer and receiver by 5MHz or more mitigates the effects of interference substantially; moreover, other software techniques such as changing packet sizes, rates, modulations, CCA thresholds and modes, and adding FEC were shown in Section 5 to be ineffective.

Note that rapid channel hopping has been applied in Frequency Hopping Spread Spectrum (FHSS) systems such as the original 802.11 standard and Bluetooth. The novelty of our design lies in combining *rapid* channel hopping at the driver-level with Direct Sequence Spread Spectrum (DSSS) at the physical layer at the same time. There are two main reasons to combine both channel hopping and DSSS techniques in this manner. First, the vast majority of commodity NICs today support DSSS in hardware, and the ability

to change channels in software. Second, physical-layer frequency hopping of the sort done by FHSS and Bluetooth imposes significant timing constraints on the hardware, and restricts them to low data rates (2Mbps in the case of 802.11 FHSS and 3Mbps in the case of the latest Bluetooth 2.0); so, high-rate physical-layer frequency hopping designs are unlikely to emerge in the near future.

CH can be effective because, as shown in Section 4.5, an interferer occupying an immediately adjacent channel does not cause significant performance degradation. Another way to take advantage of this observation is to use channel switching, such as is done by 802.11h for mitigating 802.11 co-channel interference. However, we believe that a robust design needs to hop channels *rapidly* in order to avoid other frequency hopping interferers such as cordless phones and privacy jammers. In addition, the required hopping rate is closely related to the speed with which malicious frequency hopping interferers can switch channels and the number of channels available. The 802.11 network can only use a channel for as long as it takes for an interferer to discover the channel being used. Since there are 11 channels in 802.11b/g and a PRISM NIC based attacker can switch channels in $250\mu\text{s}$, the dwell time on a channel can only be a few ms long (i.e., around $11 \times 250\mu\text{s}$).

6.1 Design and Implementation

We first describe the key goals and challenges that drove our design, and then briefly describe our implementation.

Our design has two main goals. First, it must be efficient and withstand even malicious interferers. As a result, it must balance the channel dwell period, during which it can actively use a channel, with the channel switching latency. Second, we should be able to implement it on commodity NICs without changing their MAC or PHY.

The combination of these two considerations leads us to use a dwell time of 10ms. The hardware-imposed channel switching latency of PRISM NICs is $250\mu\text{s}$ and it is less than $500\mu\text{s}$ for Intel NICs. In our implementation with PRISM NICs, this results in a reasonably low 2.5% overhead. Further, during periods of interference, a node will defer packet transmission by up to 2.5ms due to carrier sense (for a contention window of 127 slots with a slot duration of $20\mu\text{s}$). Since the dwell time is only 10ms, and since each packet is retried up to 7 times, these lengthy deferrals will ensure that packet loss is minimized during dwell periods when there is heavy interference.

To ensure resistance to attackers, only legitimate users should know the channel hopping sequence. We accomplish this by using an MD5 hash chain to decide the next channel to hop to. Starting with an initial seed, we repeatedly hash the current value, extract the lower four bits, and use them to decide the next channel to jump to, as follows. If their value is between 1 and 11, we use the value as the channel number. If not, we discard the bits, and compute the next MD5 in the hash chain. The resulting sequence will be pseudo-random

and cryptographically strong. All legitimate nodes can compute this chain as long as the nodes agree on a value in the hash chain at some point. Assuming that the interferer is outside the network, the network can use WEP or WPA based encryption to securely exchange this hopping information.

To minimize implementation issues, we try to avoid any global synchronization and non-backward compatible changes to 802.11 control messages or MAC behavior. In normal operating conditions, the AP does not perform channel hopping. However, as soon as the AP detects link degradation, it creates a MD5 seed and starts hopping. As a result, all clients immediately become disconnected. The reaction to this disconnection is that each client begins scanning all channels for an AP from the network. This reaction is the standard behavior in current 802.11 implementations. Eventually (in a few seconds), each client synchronizes with the hopping AP on some channel by successfully transmitting a probe request and receiving a probe reply. Thus, this synchronization is a one-time cost for a client. The probe reply contain the AP's current encrypted MD5 value in the "Information Elements" section (this section is designed to be extensible). To further simplify our implementation, we do not provide any special error handling during channel switching. The 802.11 MAC works unchanged within a dwell period. During channel switching, the MAC on the NIC can be made to not transmit packets. The receiver is likely on the same channel as the transmitter before and after channel switching, but our implementation does not guarantee this because channel switching is triggered by the driver and not directly by the NIC. We rely on 802.11's built-in retransmission facility to handle any missed transmissions. An interesting question is what happens when the interference is mostly localized at the client and not at the AP. Theoretically, the client could trigger the above procedure at the AP, but we have not implemented it.

Given this design, the most effective attack is for an adversary to cause three successive beacon losses, which results in a disconnection event as described in Section 4.1. To estimate the probability of such an attack succeeding, we assume that beacons are transmitted every 100ms (default in most APs), and that APs jitter these beacon transmissions by up to 10ms (the channel dwell time). The best strategy for a malicious interferer is to randomly pick a channel, blindly disrupt it for a short period, pick another random channel and repeat the attack. This strategy is optimal since it allows the malicious interferer the highest duty cycle for attacks. Any other strategy, such as first listening for transmissions on the channel before attacking, incurs an additional delay that can be several milliseconds long. Also, if the attacker remains stationary on a channel, the 802.11 network can avoid such channels by using 802.11h-like extensions. However, even the optimal attack only has a probability of $\frac{1}{11}$ of generating interference because there are 11 802.11b/g channels. This means the probability of three successive beacon losses is less than 0.1%. This calculation overestimates the probabil-

ity of three consecutive beacon losses because, in practice, beacon transmissions can be deferred by the AP until the attacker moves away to another channel.

We implemented a Channel Hopping prototype using PRISM NICs. We use the low-level PHY interface described in Section 4.1 in order to switch channels. We modified the *hostap* driver to switch channels every 10ms. Our implementation uses only one NIC at the AP, and is therefore susceptible to lost packets when clients are not on the same channel as the AP. This may result from unsynchronized channel hopping due to clock drift. It is possible to eliminate this problem by using two NICs at the AP, one of which listens to the old channel while the other uses the new channel. We do not change any 802.11 parameters, such as CCA thresholds, since our measurements (Section 5.2) show that such changes are not particularly helpful and that they can lead to adverse side-effects such as unfairness.

6.2 Evaluation

This section describes our experimental setup and presents the main results from our evaluation.

Our experimental setup consists of an AP (*AP*), three clients (*C1–C3*), and three PRISM interferers (*P1–P3*) that are all suspended from the roof of a large office floor building. The clients, the interferers and the AP are StrongARM-based embedded Linux boards [4]. UDP and TCP transfers occur between *AP* and *C1*. We use clients *C2–C3* in order to ensure rendezvous works with multiple clients during CH, and to verify we obtain qualitatively similar results using them instead of *C1*. There are also three ground-level interferers in the form of a cordless phone, a Zigbee sensor mote, and a video camera jammer.

Our PRISM NICs only supports 802.11b. We observe a link throughput of 4.4Mbit/s from *AP* to *C1* during unidirectional UDP transfers (1500-byte packets) with no channel hopping. With channel hopping but without interference, this throughput degrades to 3.6Mbit/s. This throughput difference is attributable to the fact that our implementation does not prevent the NIC from transmitting packets immediately before, during, and immediately after switching channels. We believe that this issue can be addressed in a future implementation. These are the baseline numbers we use to measure degradation between *AP* and *C1*.

As we argued earlier, the best strategy for an interferer is to channel-hop and target the current channel used by the network. Unfortunately, current PRISMs cannot sense the medium for several milliseconds after launching an interference pattern because of RF settling time issues. To overcome this artifact, we give each PRISM NIC access to an oracle. Once an interferer has selected a random channel, it queries the oracle whether the channel is being used, and the oracle replies with a yes/no answer within 1ms. We chose a 1ms delay because it is the minimum RF turn-around time between continuous-wave interference and sensing that we found on devices ranging from 802.11 NICs to Zigbee to Bluetooth.

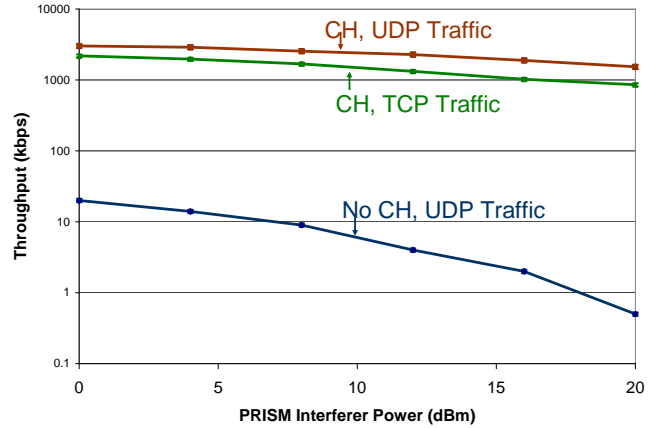


Figure 11—Throughput vs. interferer power with and without CH.

We first measure the performance of CH with a single PRISM interferer *P1*. Figure 11 shows the impact that increasing the transmission power of a single PRISM interferer has on the throughput between the AP and a client. We control the power of the PRISM interferer in software from 0dBm to 20dBm. For the CH lines, both the network and the interferer hop channels. We show the link performance using UDP and TCP traffic (TCP connections stalled and throughput dropped to zero without CH). Note that throughput is shown on a log-scale. The plot also contains confidence intervals for all data points, all of which are within 6% of the values of the data points (they are once again too small to see clearly because the throughput scale is logarithmic). With CH, UDP throughput in the presence of a 0dBm interferer is 3Mbps, which is about 68% of the baseline interference-free channel and 83% of an interference-free channel that uses CH. This is two orders of magnitude better performance than a network that does not channel-hop under interference. TCP, which is more susceptible to interference-related delays and losses, obtains throughput of about 70% of UDP under interference. This illustrates that CH enables both TCP and UDP performance to gracefully degrade as interference increases.

To obtain a deeper understanding of how CH reacts to interference, we measured the transmission behavior and latencies of packets. For each interferer power level, Figure 12 plots the percentage of packets that were successfully transmitted in the first try, those that needed a single retry, those that needed multiple retries, and those that were discarded, and, therefore, lost. The plot also shows the average packet latency across all transmissions for each power level. Note that the average loss rate is small, less than 4% even with heavy interference. This rate is less than the channel overlap probability between the interferer and the network ($= \frac{1}{11}$) because the transmitter’s MAC defers transmitting a packet for some time when the interferer is active, and because there can be up to 7 retransmits per packet. The average latency is under 200 μ s in all cases, and increases gradually as the in-

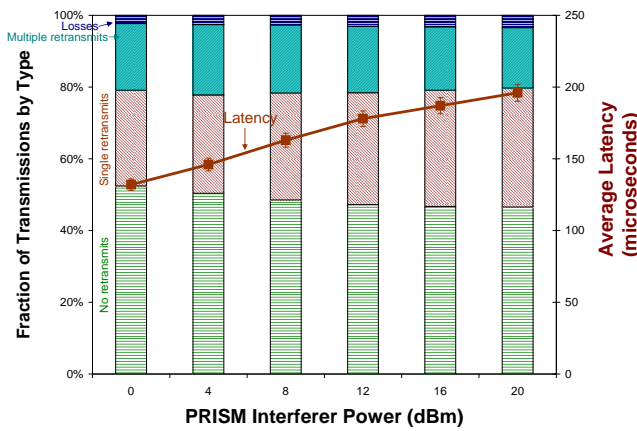


Figure 12—Transmission types and average latency vs. interferer power with CH.

terferer power increases. Latency increases mainly because of deferrals and losses during those dwell periods that it encounters the hopping interferer. As can be seen from the graph, increasing power causes a few less packets to reach successfully on the first try. However, these transmit errors are largely recovered by a single retransmission, and hence, the slight increase in packets requiring a single retry. Note that the percentage of packets that need multiple retransmissions remains fairly constant across the interferer power.

We next measure the performance of CH by increasing the number of interferers. We launch up to three unattenuated PRISM interferers, $P1-P3$, whose transmit powers are the same as the 802.11 network, and who coordinate their interference schedules so that they do not overlap—the optimal strategy for three transmitters. These interferers can therefore occupy all three 802.11b/g orthogonal channels at least part of the time. We should note that 11 interferers could occupy all channels simultaneously. However, this requires more hardware and we do not consider it here. We also add other interferers to the PRISMs, in descending order of their interference capability: a video camera jammer, a Zigbee sensor node, and a cordless phone.

We show the throughputs and latencies for each configuration in Figure 13. Throughput is plotted linearly this time. We can see that, even with heavy interference, the UDP link throughput stays above 600kbps. It drops almost linearly with the number of PRISM interferers, and more gradually as we add other interferers. The more gradual decrease is because these interferers are narrow-band. So, unless the interferer happens to fall squarely within the current channel, we can use the channel during the entire dwell period, without delays or losses. We also measured throughput under TCP and it is 20%–40% worse than under UDP. In addition, the PRISM interferers impact the TCP transfers more heavily due their ability to create higher latencies and losses than the other interferers. The average packet latency across every transmitted packet is also shown in Figure 13. It also

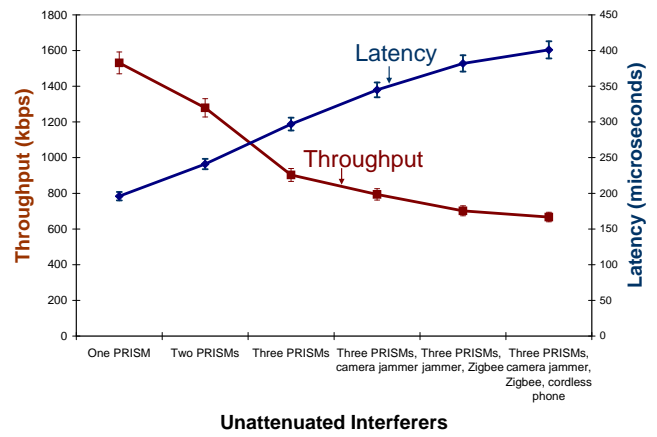


Figure 13—Throughput and latency vs. interference with CH.

follows a similar trend as the throughput curve and reaches a maximum value of about $400\mu s$. We measured the loss rates in each case and found them to be under 5%. This loss rate is not much different from the single interferer scenario shown in Figure 12. This is because rapid channel hopping quickly finds time slots during which there is light interference, and because deferrals and link-local retransmits retain a packet until such a usable time slot can be found. These results show that CH can withstand large amounts of interference well and ensure graceful degradation.

7. RELATED WORK

There are three main bodies of relevant prior work. The first category deals with RF interference and jamming, the second with Denial-of-Service (DoS) in 802.11, and the third with channel hopping. We consider each below.

RF interference and jamming. RF interference is a classical problem in signal processing, and much prior literature on theoretical and simulation studies of narrow-band jamming on 802.11 modulations exists [21, 26]. While they highlight specific vulnerabilities with particular modulation schemes, they mainly consider the demodulator performance in isolation. In this paper, we show that commodity NICs are vulnerable to interference at several additional points in the receiver processing path.

Spread spectrum techniques such as DSSS and FH can inherently withstand jamming [27]. DSSS is better at withstanding wide-band interference, while FH is better at withstanding narrow-band or pulse jamming, which can be caused by selfish devices such as Zigees. Also, while 802.11b uses DSSS, it uses the same spreading code on all 802.11b devices, which renders it ineffective against other DSSS interferers, such as malicious PRISM devices. Thus, adding CH, which is a form of FH, to 802.11b/g can significantly mitigate both selfish and malicious interference.

The 802.11 standard includes several task groups [11] for minimizing interference and ensuring interoperability among 802.11 devices, and between 802.11 and specific non-802.11 devices such as military radars. For example, 802.11f spec-

ifies how multiple APs of the same network can coordinate to support client hand-offs and roaming, and 802.11h specifies channel energy measurement techniques in order to detect and avoid military radar interference in the 5GHz band. Unfortunately, 802.11 currently does not include protocol-level coexistence with non-802.11 protocols beyond the basic CSMA/CA mechanism. As we saw, this is insufficient to handle selfish and malicious devices. \square

802.11 DoS. Several researchers have highlighted 802.11's vulnerabilities against DoS attacks. Bellardo *et al.* [16] show how one can launch DoS attacks against 802.11's management and media-access protocols, and Ferreri *et al.* [18] describe DoS attacks against an AP's association and authentication mechanisms. Such attacks purely target the MAC layer, while our focus is largely on PHY layer interference. Glass *et al.* [19] describe a DoS attack on both the MAC and the PHY layers. At the MAC layer, they use techniques similar to those in [16], and at the PHY layer, they introduce interference to study CCA deferrals at the transmitter. Similarly, Wullems *et al.* [7] also study CCA deferrals. In contrast, we mainly consider losses at the receiver because these losses are the key contributors to severe degradation seen in practice.

Channel Hopping. A recent system design, called SSCH [15], uses channel hopping to improve the capacity of multi-hop ad hoc networks. SSCH uses channel hopping to prevent interference between simultaneous transmissions at adjacent nodes. SSCH proposes a slot time of 10ms but does not include an implementation for rapidly switching slots. Similarly, Mishra *et al.* [24] explore the use of channel hopping to improve the fairness and performance of overlapping 802.11 network deployments. Their system, called MAXchop, uses slow channel hopping rate of around one second per hop, and, thus, does not address malicious interference.

8. CONCLUSIONS AND FUTURE WORK

We showed that selfish and malicious interferers cause substantial interference degradation in commodity NICs. We identified several NIC-related vulnerabilities that can be exploited to allow even attenuated and narrow-band interferers to be highly effective. We proposed extensions to the SINR model that capture two main limitations of such NICs, namely, limited dynamic range selection and non-linearity in receiver sensitivity. We showed that this extended SINR model can accurately predict that changing 802.11 operational parameters would be ineffective at reducing the impact of interference and that channel hopping could be helpful. We designed and implemented a rapid channel hopping scheme using commodity NICs, and showed that it greatly improves interference tolerance while incurring little additional overhead. While this work was done in the context of 802.11b/g networks, we believe that the lessons about how commodity NIC designs are vulnerable at much lower SINR levels than had been expected and how to address this issue are more widely applicable. We plan to extend our work to other link-layer technologies in the future.

References

- [1] Bluetooth coexistence with 802.11, <http://tinyurl.com/2grndv>.
- [2] ANT Personal area networks, <http://www.thisisant.com/>.
- [3] PRISM Driver Programmer's Manual, <http://tinyurl.com/yrqh27>.
- [4] The Stargate Platform, <http://tinyurl.com/2b1d2v>.
- [5] Wii interference with 802.11, <http://tinyurl.com/y4gyvv>.
- [6] WirelessUSB from Cypress Semiconductors, <http://tinyurl.com/ywjo8s>.
- [7] Denial of Service Vulnerability in IEEE 802.11 Wireless Devices, <http://tinyurl.com/26x94r>.
- [8] CC2420 Data Sheet, <http://tinyurl.com/ytut3k>.
- [9] Wireless camera jammer for privacy, <http://tinyurl.com/2esyv4>.
- [10] Citywide wireless broadband projects around the world, <http://www.muniwireless.com/>.
- [11] Quick guide to IEEE 802.11 WG Activities, <http://tinyurl.com/ypojvc>.
- [12] A Brief Tutorial on the PHY and MAC layers of the IEEE 802.11b Standard, <http://tinyurl.com/2d6f48>.
- [13] PRISM Data Sheets, <http://tinyurl.com/ywr992>.
- [14] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in chaotic wireless deployments. In *MobiCom '05*.
- [15] P. Bahl, R. Chandra, and J. Dunagan. SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *MobiCom '04*.
- [16] J. Bellardo and S. Savage. 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In *Usenix Security 2003*.
- [17] J. Eriksson, S. Agarwal, P. Bahl, and J. Padhye. Feasibility study of mesh networks for all-wireless offices. In *MobiSys 2006*.
- [18] F. Ferreri, M. Bernaschi, and L. Valcamonici. Access points vulnerabilities to dos attacks in 802.11 networks. In *WCNC 2004*.
- [19] S. Glass and V. Muthukumarasamy. 802.11 DCF denial of service vulnerabilities. In *3rd Australian Computer, Network & Information Forensics Conference*.
- [20] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *MobiCom '03*.
- [21] T. Karhima, A. Silvennoinen, M. Hall, and S.-G. Haggman. IEEE 802.11b/g wlan tolerance to jamming. In *MILCOM 2004*.
- [22] M. Kodialam and T. Nandagopal. Characterizing achievable rates in multi-hop wireless networks: the joint routing and scheduling problem. In *MobiCom '03*.
- [23] M. Lacage and T. Henderson. Yet another network simulator. In *WNS2'06: Proc. of the 2006 workshop on ns-2*.
- [24] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee, and S. Ganguly. Distributed channel management in uncoordinated wireless environments. In *MobiCom '06*, .
- [25] A. Mishra, V. Shrivastava, S. Banerjee, and W. Arbaugh. Partially overlapped channels not considered harmful. In *SIGMETRICS '06*, .
- [26] J. Park, D. kim, C. Kang, and D. Hong. Effect of partial band jamming on ofdm-based wlan in 802.11g. In *ICASSP 2003*.
- [27] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR.