Interconnecting WiFi Devices with IEEE 802.15.4 Devices without Using a Gateway

Shengrong Yin, Qiang Li, Omprakash Gnawali
Department of Computer Science
University of Houston
{syin, qiangli, gnawali}@cs.uh.edu

***Abstract*—In many wireless sensing and control application deployments, there is often a gateway device to bridge between the low power IEEE 802.15.4 network and the Internet. The bridge has at least two interfaces. One interface communicates with the 802.15.4 wireless. The other interface either communicates with WiFi or wired network. When a user wants to send a command to the wireless controller, lets say at a smart home, the user may use a smartphone and send command over WiFi to the gateway, often through a cloud service provider. Then gateway shuttles the message from the wired or WiFi chip to the 802.15.4 chip. Then the gateway transmits the messages over the 802.15.4 chip into the 802.15.4 network. In this work, we design a novel modulation technique that runs on the WiFi devices (e.g., smartphone) and demodulation technique that runs on 802.15.4 devices (e.g., a wireless controller in a smarthome) to enable WiFi devices to directly communicate with 802.15.4 devices without any gateway. The key idea is to utilize crosstalk between 802.11 and 802.15.4 channels as the medium for communication. We implemented the proposed technique on multiple platforms and are able to successfully achieve a data rate of 2 bytes per second with less than 10% bit error rate in uncontrolled environments.**

***Keywords*—crosstalk, 802.15.4, RSSI, 802.11**

I. Introduction

Wireless sensing and control applications are increasingly being deployed in our homes and environments to enhance comfort for the occupants [1], understand activities and energy use in a home [2], [3], increase energy efficiency [4], and allow better automation and control [5]. Many of these applications require users to interact with the sensor or control devices. For example, the user may want to control the light or thermostat in the house. The user may use a smartphone to perform such control actions. The control actions are conveyed to the wireless sensors or controls through the Internet. Some smart home automation applications are non-interactive. Yet, they require Internet access either to upload the data or download configuration information. Thus, in many scenarios, the wireless sensor and control devices in a smart home, office or environment require communication to or from the Internet.

The existing solution to enable such a communication is through a gateway or a bridging device. At a high level, the device is a router. It has one 802.15.4 interface to communicate with the 802.15.4 network. The other interface may be WiFi or wired. The device shuttles traffic between the two interfaces. A control message (e.g., to turn a light on) coming from a smartphone app, travels to the gateway (possibly through the Internet), is translated appropriately for 15.4 network, and is transmitted by the 15.4 radio. On the software side, the bridging may happen at the application layer (with custom application-specific messages) or at the network layer (with standardized network layer protocols). Recent IETF standards such as 6LoWPAN [6] support development of sensor networks with this architecture. This architecture, which we call gateway-oriented architecture, has served us well as evidenced by vibrant ecosystem of smart home devices and companies that sell those products. Despite some application deployments using WiFi-based sensors and controllers, 802.15.4 or low-power low-rate radios occupy a unique point in the price and design space that they are likely to be a radio of choice for many years to come.

In this paper, we challenge the premise behind the gateway-oriented architecture: that to enable WiFi devices to send messages to 15.4 devices, we need to build a gateway with the two interfaces. While modern gateway devices provide additional functionalities such as local storage service, the gateways that ship with the 15.4 devices are primarily used to bridge between the Internet and the 15.4 network. We propose to eliminate the gateway from the network and enable WiFi devices to directly communicate with the 15.4 devices. If this is possible, we would significantly simplify the deployments and reduce the device and maintenance cost of the networks.

The core idea in our approach is to have WiFi devices transmit packets with special patterns representing the information to be conveyed to the 15.4 network. The transmission is done on a WiFi channel that overlaps with the 15.4 channel on which 15.4 devices are listening. The 15.4 devices sample the signal on the channel due to WiFi transmissions (which we would typically call crosstalk or interference and try hard to avoid) and interpret the information in the pattern. We call this technique *crosstalk-based communication (CTC)*.

Building such a modulation and demodulation scheme to enable communication from WiFi devices to IEEE 802.15.4 devices using crosstalk has two main challenges. First, WiFi channels and IEEE 802.15.4 channels are allocated for different frequencies, though the frequency bands overlap partially. Transmissions on the overlapping channels result in crosstalk and interference rather than communication of data. Second, direct communication requires both the devices to perform modulation and demodulation, compared to the gateway-oriented solution, in which the gateway does the modulation or demodulation using the radios designed for the specific frequency band. The demodulation, especially on the 15.4 devices has to be efficient in both power and computation. Any system we design must not only overcome these challenges but also offer at least a modest but usable data rate, for example, sufficient for device configuration or commands.

We have designed and implemented the proposed system on multiple WiFi devices (laptop with WiFi interface, and an OpenWRT-compatible wireless AP) and on two mote platforms (TelosB and Opal). We find that the proposed technique can be used to successfully send messages from WiFi devices to 15.4 devices. Even in uncontrolled environments with other APs and Bluetooth in a residential environment, we were able to achieve a data rate of up to 2 bytes per second with less than 10% bit error rate.

We make these contributions in this work:

- We present the first crosstalk based primitive to enable communication between WiFi devices and IEEE 802.15.4 sensor nodes without a physical gateway. The primitive is a novel modulation scheme that runs on WiFi devices and demodulation scheme that runs on the 15.4 devices.
- We implement the proposed technique on real WiFi and 15.4 devices and perform experimental validation of the techniques in both controlled anechoic chamber and in uncontrolled environments. We achieve a data rate of 2 bytes per second with less than 10% bit error rate in uncontrolled environments.

II. Related work

We briefly review work related to Internet connectivity to sensor networks and study of cross technology issues in wireless networks.

Connecting to the sensor and control devices from the Internet. Most interesting and useful sensor network and control applications require them to be connected to the Internet for configuration or data access. Some sensor networks use WiFi radios. These networks directly connect to the Internet. Many sensor and control networks use low power radios, such as the 802.15.4-compliant radios. Gateway devices are typically used to bridge those networks. There has been two major efforts on this front. The first and slightly outdated method uses various application or other types of gateway devices built over serial, USB, or Ethernet hardware interface to a gateway device. Classic TinyOS serial forwarder protocol is an example of this approach. A more modern approach is to use a standardized protocol, such as 6LoWPAN [6]–[9], over the serial or other interface, so the gateway essentially becomes a network-layer routing device. Regardless of the layer at which the message switching occurs, the gateway device needs a 15.4 radio and a wired or WiFi interface where Internet devices may connect. On research projects, it is common to connect a TelosB [10] or other mote to the computer and use the computer as a gateway between the Internet and the sensor network. In commercial products, the gateway often is a standalone device that connects either to the home router by Ethernet cable or by WiFi. Chebrolu et al. investigated the feasibility of the unidirectional communication from 802.11 devices to 802.15.4 devices in [11]. But no system implementation or evaluation has been conducted based on their experience.

In our work, we design and implement a technique to connect to the sensor and control devices from the Internet without using a separate physical gateway.

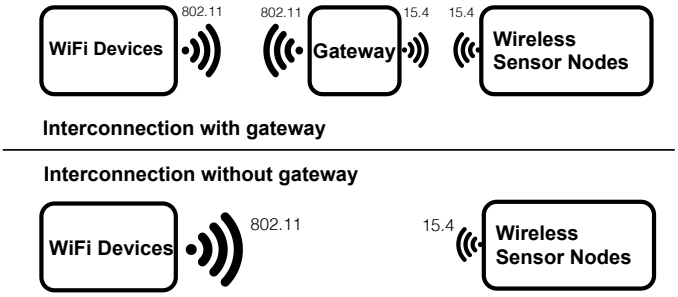

Fig. 1. Difference between the prevalent approach that uses the gateway device and the proposed approach that does not use the gateway devices for WiFi devices to communication with the wireless sensor nodes.

Wireless Interference. There has been a large body of work in understanding wireless interference, either within sensor networks or cross-technology interference. Gollakota et al. presented a decoding methodology to make 802.11n network robust under the presence of high power cross-technology interference in [12] . The system can decode messages even when receiving interfering signals from other technologies, allowing devices from different technologies to coexist. Hithnawi et al. presented a real time approach to detect and mitigate cross-technology interference in [13] . Hauer et al. introduced an interference detector which was capable to distinguish different types of interference as well as WiFi beacons in [14]. Hermans et al. also presented a system which can detect different interferers by observing the disrupted 802.15.4 packet in [15]. Hauer et al. investigated how to estimate bit error positions in a corrupted packet based on RSSI temporal variations in [16]. All the listed papers here assume WiFi activity can corrupt bits in a 802.15.4 packet and design techniques to survive from such interference [17], [18]. There is another body of work that tries to understand the performance of links on different channels [19], [20]. Many such studies empirically studied the performance on channels that also overlap with WiFi thus quantifying the negative impact of WiFi traffic on packet transmission performance on the 15.4 links. In our work, rather than looking at interference and crosstalk as a nuisance, we use it to enable communication between WiFi and 15.4 radios.

New Wireless Communication Channels. Recently, new types of wireless channels have been developed for use in sensor networks. For example, Liu et al. presented a design for communication using only ambient RF, by backscattering the ambient RF [21], [22]. There are also interesting work on developing Visual Light Communication channels for communication in wireless sensor networks. For example, Giustiniano et al. and Wang et al. created a visual light communication system with a fully functional Linux-based PHY and MAC layer implementation [23], [24]. Rajagopal et al. enabled light communication for low power embedded devices by utilizing cameras on consumer devices [25]. They achieved a data rate of 1.25 bytes per second. These are examples of research developing new medium for wireless communication. In a similar spirit, in this paper, we design and implement CTC between 802.11 and 802.15.4 by utilizing crosstalk between the two technologies.

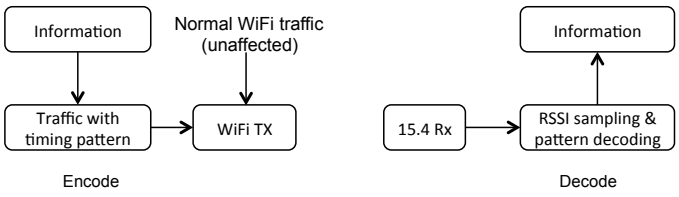

Fig. 2. Components of the proposed communication system that utilizes crosstalk between 802.11 and 802.15.4 channels.

III. System Overview

In this section, we present the design of our system that allows direct communication from a WiFi device to 802.15.4 networks without using a physical gateway device.

A. System Architecture

Our goal is to allow WiFi devices to send messages to the devices that use the 802.15.4 radios. Thus, the users of our system consist of devices in these two networks. First, the devices that operate in 802.11 networks. For example, iOS and Android based phones, the wireless adapters used in the laptop, or wireless access points operating in 2.4 GHz frequency band. Second, we also have the platforms deployed in 802.15.4 networks. These are typically low power devices with transceivers operating in 2.4 GHz frequency domain. Examples of such devices include TelosB as research platforms or smart gadgets in smart homes. As shown in Fig. 1, the main difference between the prevalent approach and our approach is we enable the communication between these two sets of devices without the gateway device.

The basic idea of our approach is to make use of the cross technology interference to encode and decode information. Fig. 2 shows the main components that makes this type of communication possible. Information is encoded as special timing patterns of UDP packet frames. The idea is inspired by Lee et al. 's work [26] on covert timing channel in which they control and access every bit transmitted in physical layer. Such precise timing pattern was implemented on a highly customized NICs with wired network. They created the covert channel by controlling inter-packet delays to guarantee the network security. In our work, the packets are sent over commodity WiFi interface of an AP or other wireless devices with no such precise timing control on the inter-packet delay nor any change in device drivers. The 802.15.4 receiver samples RSSI on the overlapping channel and decodes the timing pattern. The timing pattern represents the information, which is passed to the application. In the following sections, we describe each step in more details with the design nuances and tradeoffs.

B. Utilizing Crosstalk Between 802.11 and 802.15.4

Our approach takes advantage of cross technology interference that exists between the 2.4 GHz channels and the 802.15.4 channels. The WiFi transmitter does not do anything special at the physical layer to encode information using the crosstalk. At the physical layer, the transmission looks like transmission of any other packets. The 802.15.4 receiver, however, is not designed to receive packets from 802.11. So, a regular packet reception mechanism does not work. Instead, the receiver

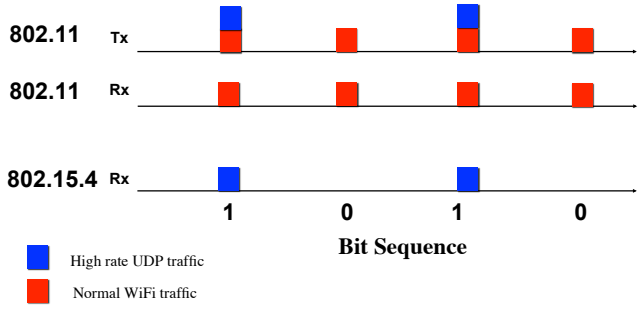

Fig. 3. Transmission from WiFi devices to IEEE 802.15.4 sensor nodes on crosstalk channels. The sensor nodes can detect the presence or absence of high rate UDP traffic on the channel even though they cannot receive the normal WiFi packets. These signals can be used to encode information. In this example, presence or absence of high rate UDP traffic on the channel is used to decode the bit string "1010".

samples the RSSI on the channel at a few KHz. The signals transmitted in 802.11 channels can be received (even though the packets cannot be decoded) in the nearby 15.4 channels (Fig. 4). Such transmissions cause the 15.4 channels to be many times saturated with the signal. This leaked signal can be detected through background RSSI sampling on the 802.15.4 transceiver. We can modulate and demodulate these crosstalk signals based on the leaked signal characteristics. For example, in our system, we modulate the leaked signal to enable the communication from WiFi devices to IEEE 802.15.4 based sensor node (Fig. 3).

C. Modulation by WiFi Devices

WiFi devices modulate the crosstalk signal to send information to the sensor nodes. The information is encoded as timing patterns (on-off). The code itself is represented by controlling the presence and absence of high rate UDP packets on the WiFi channel. The presence of high rate UDP packets is defined as One. The absence of high rate UDP packets is defined as $Zero$. For accurate modulation, the timing of the traffic patterns needs to be accurate. In a general-purpose operating system, maintaining accurate timing on the outgoing WiFi interface requires accurate timestamping. For our experimentation, we build a packet generation tool. Our packet generation tool is similar to $iperf$, but it can generate high rate UDP packets with microsecond-level accuracy.

Using the packet generation tool, we can send back to back packets to achieve a maximum packet rate of nearly 3000 packets per second. Each packet has 1500 bytes. In the best case, if we send one packet to saturate the channel (indicating a '1') and wait for one packet to indicate a '0', we can theoretically achieve a data rate of 3kbit/s with level triggering technique. However, sending one packet will only take 300 μs. This symbol rate will typically be too fast for sensor nodes to decode successfully without errors. The decoder would need to be synchronized and perform high speed channel sampling. Thus, in our system we use much lower symbol rate so even a modest sensor platform such as a TelosB or an Opal mote can decode the information correctly.

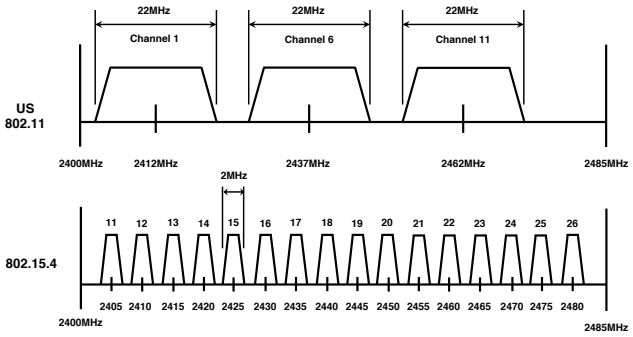

Fig. 4. Map of 802.11 and 802.15.4 channels. These two sets of channels overlap with each other and cause crosstalk.

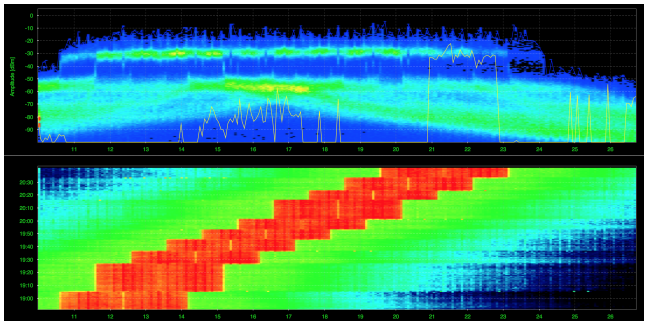

Fig. 5. Screen shot taken from Chanalyzer (a tool for visualizing wireless landscape) using Wi-Spy 2.4x tool during the measurement study in the anechoic chamber. The figure shows quiet channels other than the ones used for WiFi transmissions (red).

D. Demodulation by Sensor Nodes

We now describe how the sensor node detects the channel and decodes the information on that channel.

1) Channel Detection: There are two models for how the sensor node decides on the channel to use for reception. The first model is manual configuration. This approach is similar to how we configure many WiFi or sensor devices. For example, when we program sensor devices, we set the radio channel. Similarly, in our system, we can manually configure the sensor device to listen for messages from the WiFi network on the 15.4 channel with the largest overlap with the WiFi channel.

The second model uses automatic detection of channel. We perform several experiments to collect data and provide heuristics to detect the channel used for communication. At a high level, WiFi transmitter sends a known pattern of signals on the channel. The sensor node receiver cycles through all the channels to receive the stated pattern. To test the feasibility of this technique, we perform RF experiments in an anechoic chamber. In the experiment, we had one laptop transmitting packets back to back on WiFi channels 1-11. We use Wi-Spy [27], a portable USB spectrum analyzer, to collect the wireless signal in 2.4 GHz frequency band and visualize them with Channalyzer (Fig. 5). We also had 16 TelosB motes tuned to 15.4 channels 11-26 sampling their respective channels at 4 KHz. Fig. 6 shows the results from measurement study. It shows that whenever a device transmits on a WiFi channel, the few motes with their radio operating on the channels overlapping with the WiFi channel can successfully sample the

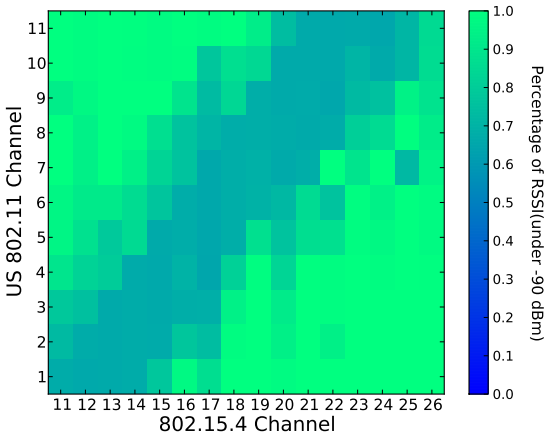

Fig. 6. Signal on the channel sampled by the sensor nodes with WiFi transmitters transmitting on all the channels in an **anechoic chamber.** Each cell represents an average from 11 rounds of 65,536 measurements.

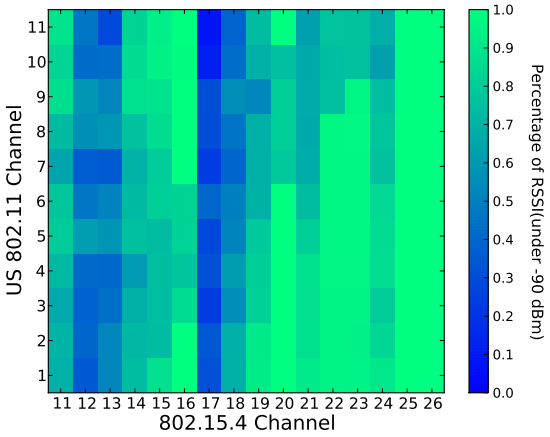

Fig. 7. Signal on the channel sampled by the sensor nodes with WiFi transmitters transmitting on all the channels in a **residential building.** Each cell represents an average from 11 rounds of 65,536 measurements.

channel and detect the signal. The other channels are relatively quiet. Thus, if we cycle through the channels when there are known signal patterns, we may be able to detect channels to be used for reception at least in a controlled or a quiet environment. In an uncontrolled environment, this heuristics will not work reliably as demonstrated by our second round of measurement studies, which we describe next.

In the second study, we repeat the same measurements but in an apartment building. There are other WiFi and Bluetooth devices and hence may bleed into the channels used for crosstalk-based communication (CTC). Fig. 7 shows the results from these measurements. Although, the pattern has some similarity to the pattern from the controlled environment, there is one important difference: the blue vertical bands indicate certain channels are saturated (from the perspective of the 802.15.4 devices) regardless of the channel used by our WiFi transmitter. This is due to WiFi routers using channels 1 and 6 in the building. In the residential apartment, Most wireless APs are operating on channel 1 and channel 6. Thus, simple

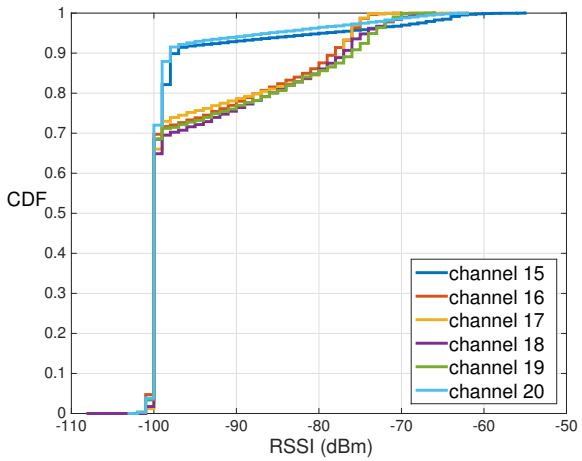

Fig. 8. CDF of RSSI on 802.15.4 channels 15-20 with WiFi transmitter on 802.11 channel 6.

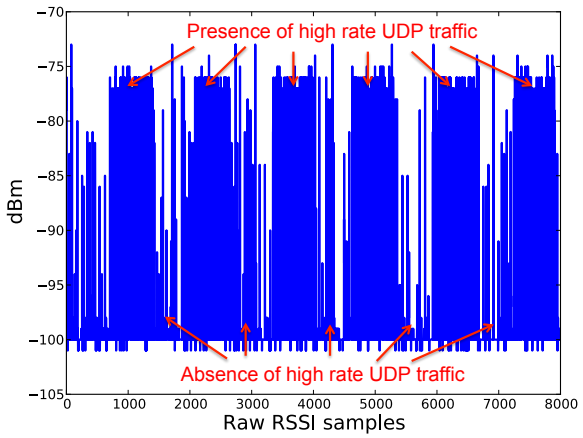

Fig. 9. Raw RSSI values sampled by a TelosB mote on channel 17 with WiFi transmission on channel 6. The spikes during the absence of the high rate UDP traffic are caused by normal WiFi usage, e.g, web browsing, video streaming.

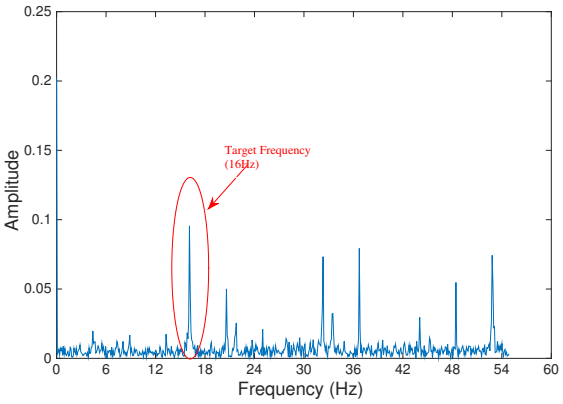

Fig. 10. FFT for RSSI traces sampled by a TelosB mote on channel 17 with WiFi transmissions on channel 6.

state-less channel scanning alone will not be able to detect the channel used for CTC in this uncontrolled setting. A robust preamble or channel detection code will be required so the sensor receiver and the WiFi transmitter converge on a channel. An alternative is to perform aggregate analysis (e.g. CDF) of the signals sampled on the channel. Fig. 8 presents the CDF of sampled RSSI from channel 15 to channel 20 from the study in the uncontrolled environment with WiFi transmitter on channel 6 in 802.11 network. We find that more than 90% of RSSI samples on channel 15 and 20 are close to -100dBm or lower. The number is 70% for channels 15, 16, 17 and 18. Thus, these four channels may be good candidates for crosstalk-based communication (CTC) with WiFi channel 6. Further measurements could narrow down the set of good channels, in this case channel 17 which has more interference. Thus, in our approach, we try to find the channels that offer the most interference (in contrast to interference avoidance work that tries to find the channels with the least interference). We have also empirically established that for manual configuration approach, we should use 802.11 channel N together with 802.15.4 channel N+11. This rule also matches the inferences shown on standard channel maps such as the one in Fig. 4.

Thus, with measurements in an anechoic chamber and an uncontrolled environment, we test the feasibility of channel scanning to find the channel for communication. We also note that manual configuration of channels is the most reliable way to synchronize the channels similar to how we configure many sensing systems today.

2) Signal Decoding: During a reasonably strong WiFi transmission, the 15.4 transmitter typically gets saturated. Thus, the RSSI samples show a pattern consisting of small values (when there are no WiFi transmissions) and high value (when there are WiFi transmissions). Fig. 9 shows a sample RSSI trace captured at 4KHz by a TelosB mote during a WiFi transmission with our encoding scheme. We observe a lot of raw RSSI spikes during the absence of the high rate UDP packet. It is due to normal WiFi traffic indicated in 2. The raw RSSI is a reflection of both the high rate UDP streams and normal WiFl streams when the wireless AP is transmitting wireless signal. We can identify the periodic on and off patterns in the raw RSSI values. Fig. 10 plots the FFT of the time series signal. The largest peak corresponds to the periodicity our WiFi-based modulation for crosstalk-based communication. This result provides evidence about the feasibility of detecting WiFi signals modulated by UDP packets with a 15.4 radio. Given the feasibility, we now design two strategies to demodulate the crosstalk signals without incurring high memory overhead.

Strategy 1: Minimum RSSI Fraction.. 802.15.4 wireless transceivers report minimum RSSI values if the received signal is below or equal to the sensitivity. Strategy 1 basically applies the minimum RSSI Fraction as an indicator to distinguish between presence and absence of high rate UDP packets.

Assuming CTC data rate and the RSSI sampling rate is known, the window size is configured to be:

$$\text{window size} = \frac{\text{sampling rate}}{\text{data rate} \times \text{sliding steps within the window size}}$$

Within each window, we first find the smallest RSSI value, which is similar to the CCA algorithm proposed in B-MAC [28]. Then, we calculate the minimum RSSI fraction over the

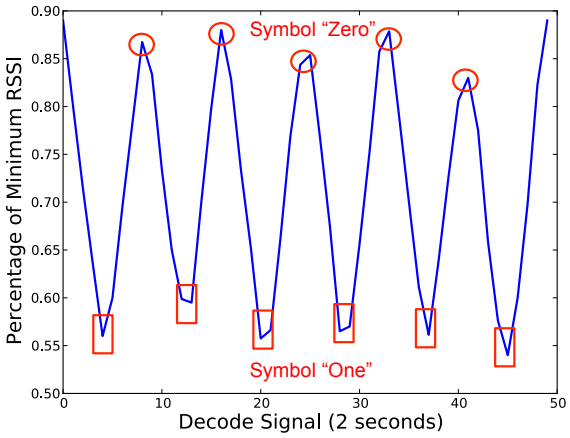

Fig. 11. Sensor node decoding the WiFi signal using the *Minimum RSSI Fraction* strategy.

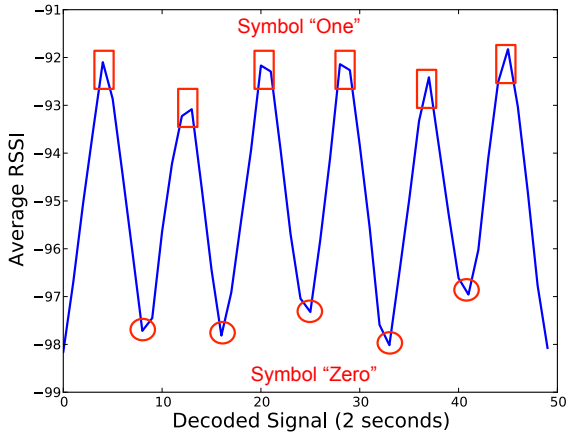

Fig. 12. Sensor node decoding the WiFi signal using the *Average RSSI* strategy.

window size. Intuitively, on a quiet channel, this fraction will be large, i.e., a symbol 0. On a busy channel, this will be small, i.e., a symbol 1. The minimum RSSI value is the constant which represents the smallest value the wireless transceiver can report. Fig. 11 shows the result decoded by this strategy. Algorithm 1 shows the details of this technique.

Strategy 2: Average RSSI.. Similar to minimum RSSI fraction, the average RSSI method also uses the same window size and computes the average RSSI for each window. Based on the average RSSI, we find the peak, i.e., "1", and the valley, i.e., "0", to decode the information. Fig. 12 shows the result decoded by this strategy. Algorithm 1 shows the details of this technique.

IV. System Evaluation

In this section, we evaluate the proposed communication technique.

Algorithm 1 Decoding Algorithm

Input: RssiSamples, WindowSize, Strategy in
Output: RssiList out

if $(Radio = CC2420)$ **then**
  MINRSSI = -101
else if $(Radio = AT86RF230)$ **then**
  MINRSSI = -91
end if
Initialization :
create queue with size equal to the WindowSize
LOOP Process
if $(Strategy = Min.RSSIFraction)$ **then**
  for $item$ in $RssiSamples$ **do**
    if queue is not full **then**
      enqueue item
    else
      minRssiFrac = queue.count(MINRSSI)/WindowSize
      RssiList.append(minRssiFrac)
      dequeue queue
      enqueue item
    end if
  end for
else if $(Strategy = AverageRSSI)$ **then**
  for $item$ in $RssiSamples$ **do**
    if queue is not full **then**
      enqueue item
    else
      avgRSSI = avg(queue)
      RssiList.append(avgRSSI)
      dequeue queue
      enqueue item
    end if
  end for
end if
return RssiList

A. Metrics and Settings

We use BER (Bit Error Rate) as the primary metric to evaluate the system reliability. We perform experiments in both residential and office-like environments since these areas are equipped with a lot of WiFi devices creating a challenging environment for our communication system. Fig. 13 shows the residential setting used in our experiment. This is an apartment with a microwave oven, a wireless AP as well as portable devices such as cellphones, tablets, laptops and several Bluetooth speakers. All these WiFi devices are connected to the wireless AP for Internet access. By experimenting in this uncontrolled environment, we can test the robustness, and reliability of the system.

B. Rate of generated UDP Packets (Packet Rate)

Our system uses generated UDP packets to modulate signals, but the artificial traffic could negatively affect the normal use of WiFi network. So our goal is to generate the traffic with the optimal 802.11 packet rate while maintaining high reliability. In order to achieve this goal, we evaluate our system in a real WiFi network scenario. We generated traffic when video streaming, web browsing, online gaming sessions were taking place by the residents. Fig. 14 shows the BER

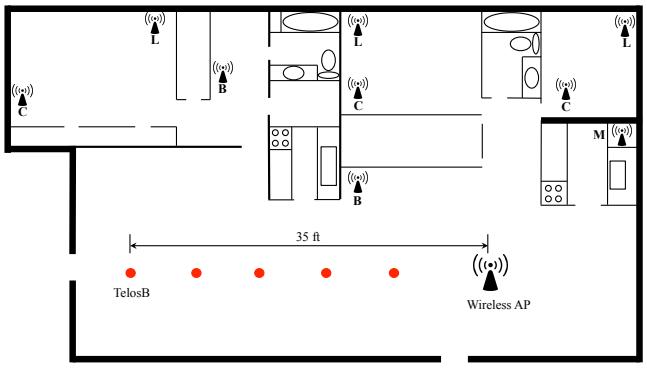

Fig. 13. Settings for experimental evaluation in a residential apartment with a few devices operating in 2.4 GHz frequency band. (M for Microwave, B for Bluetooth speakers, C for Cellphone, L for Laptop.)

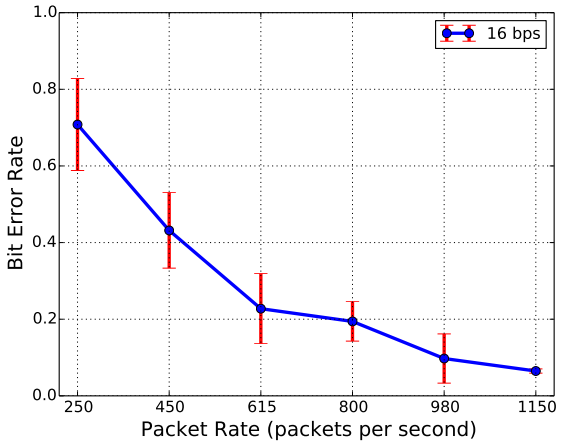

Fig. 14. BER vs UDP packet rate with a CTC date rate of 2 bytes/s

achieved at different packet rates, which is correlated with the symbol rate. During the experiment, we enable the wireless AP for Internet access, then start generating bit sequence from devices connected with this wireless AP. To control the environment settings, we allow only one associated device. We run the wireshark packet capture tool on this associated device, which is a MacBook Pro with an Intel i5 CPU. We capture all the incoming 802.11 packets on the wireless interface. We surveyed the packet rates indicated in Fig. 14. As we can see in Fig. 14, the packet rate can directly affect the system reliability. With 980 packets per second, it is possible to achieve a BER of less than 10%.

We configure the CTC data rate as 16 bits per second. As we can see in Fig. 14, as the 802.11 network traffic goes up, the Bit Error Rate significantly decreases to less than 10% and tend to be near 0%. However, the 802.11 network traffic can have has a major influence on the network performance of WiFi network users. We even tested the network performance when setting the packet rate up to 1600 packets per second. We recommend 1000 packets per second as the optimal rate for the UDP packet generated by our system. . Compared to the BER in lower packet rate, this setting provide stability and high decoding accuracy.

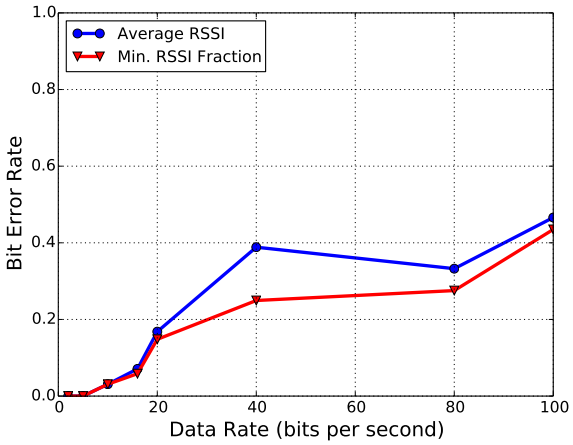

Fig. 15. BER vs CTC data rate using different decoding strategies.

C. RSSI Sampling Rate

The sensor node samples the channel to interpret the symbols. Lower sampling is less costly in hardware resources and energy. Higher sampling rate makes the system more robust and potentially allows higher data rate but at hardware or energy cost. We performed experiments with different RSSI sampling rates on the motes under two settings to determine the best sampling rate. The first setting is called *WiFi with Internet Connection*, in which case the WiFi AP had normal WiFi users performing browsing and other activities. The second setting is called *WiFi without Internet Connection*, in which case we unplugged the uplink cable from the WiFi AP and thus AP provided only local connectivity with no Internet access. Table 1 shows the results from our experiments in these two settings. We find that under *WiFi without Internet Connection*, low sampling rate, e.g., 2 KHz, is sufficient to achieve a low BER. With normal WiFi traffic (*WiFi with Internet Connection*), we needed a higher sampling rate to achieve a low BER. Overall, higher sampling rates are better when the WiFi AP is serving other normal WiFi users and also modulating the information for the CTC.

TABLE I. **BER IN TWO SETTINGS: WIFI WITH AND WITHOUT INTERNET CONNECTION.**

Internet Connection	2kHz (avg.)	2kHz(std.)	4kHz(avg.)	4kHz(std.)
No	2.71%	0.71%	2.18%	1.74%
YES	17.15%	1.59%	10.51%	1.69%

D. Decoding Strategies

We evaluated the reliability of the crosstalk-based communication (CTC) with different decoding strategies. We modulated the high rate UDP traffic into 5 continuously increasing data rate. For each modulated data rate, we decode them with different strategies to evaluate their performance. Fig. 15 presents the BER of the two decoding approaches under different rates at which the WiFi device sends information to the mote using CTC. For CTC data rate less than 16 bps, the two approaches have almost the same bit error rate, which is near 0. However, for CTC data rate that is larger than 20 bps,

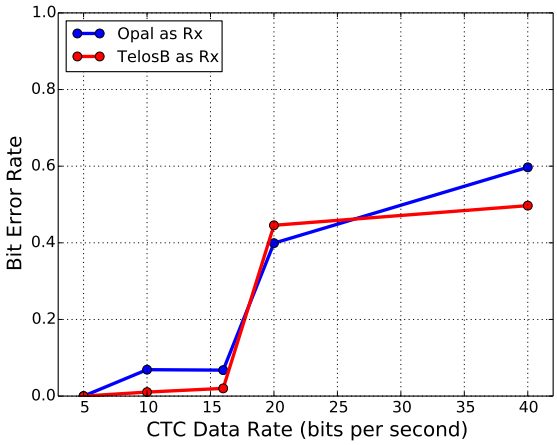

Fig. 16. BER vs data rate using different sensor nodes.

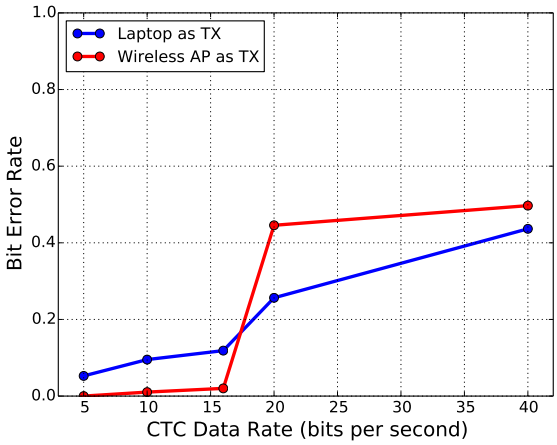

Fig. 17. BER vs data rate using different WiFi devices.

using the *minimum RSSI Fraction* decoding method results in lower BER.

E. Platform Independence

Next, we evaluate if our system works on multiple platforms both on the 802.11 and 802.15.4 networks. For 802.11 network, we test our CTC system on wireless AP and a laptop. For sensor nodes, we test our CTC system on TelosB and Opal motes. We connect both TelosB and Opal motes to a 10 port USB hub which is connected to a laptop. We programmed both the platforms with an application that samples the RSSI at 4 kHz. The motes sample RSSI and save them to the local flash. We later send this data to the laptop for data analysis. Fig. 16 compares the decoding performance for different platforms. For CTC date rate up to 16 bps, Opal provides communication with BER less than 7% while TelosB provides more reliable communication with BER less than 2%. However, increasing the CTC data rate to more than 16 bps causes the BER to become unacceptably high.

In the next experiment, we use a laptop as our WiFi transmission device. We run our packet generation tool on this laptop generating UDP packet patterns that encodes the information we want to transmit using CTC. The destination IP of these unicast UDP packets was set to be another laptop associated with the same wireless AP. While the destination IP of these packets is another laptop, the motes are able to decode information embedded in the patterns of these UDP packets using CTC. We compare the CTC data rate achieved by our system when there are other active normal WiFi users in the network and plot it in Fig. 17. We find that the CTC is more reliable if the modulation is conducted by wireless AP. Our guess is due to APs being specialized hardware for WiFi packet reception and transmission, they provide better control in timing and signal strength in packet transmissions.

F. Multiple Interferers

We now evaluate the system by exposing the system to different interferers such as Bluetooth and WiFi traffic due to different applications that may be used by other normal users of the AP. We experiment with three groups of interferers. The first is Bluetooth audio streaming. In this case, we used bluetooth to connect keyboard, magic mouse and and JBL Bluetooth speaker indicated in Fig. 13 as B (Bluetooth Devices) to MacBook Pro indicated in Fig. 13 as L (Laptop). We use CTC data rate of 16 bps and change the UDP packet rate to evaluate BER as a function of rate at which our system generates the UDP packets. In the second experiment, we use Bluetooth and YouTube streaming simultaneously on the laptop. In the third case, we use Bluetooth, YouTube streaming and a 3GB file downloading on the laptop to understand the robustness of the system under strong interference. In all these experiments, we used the wireless AP as the WiFi transmission device. We use three TelosB motes as the receiver in the sensor network. Fig. 18 shows the result of our experiment. It is worth noting that during file downloading, the WiFi nominal bit rate is always automatically adjusted by the AP. During the experiment, when we changed the UDP packet rate, we noticed that the Bluetooth speaker experienced serious time lags, which disappeared after some time. Under all circumstances, the communication achieved 10-15% BER with the highest UDP packet rate of 1800/s. Since this is a very challenging environment, in which even a WiFi-WiFi or 15.4-15.4 communication would experience a lot of losses, it is not surprising that, with smaller UDP packet rate the CTC BER goes up to 35%. Thus, we find that with appropriate modulation rate, even under heavy interference, the crosstalk-based communication system can achieve less than 10% BER.

G. Communication Range

Next, we evaluate the performance of crosstalk-based communication (CTC) at different distances. For CTC to be useful in practice, it must work at moderate distances. For example, a tablet may need to send a command to a smart device at the home. When the user carries the tablet with her to a different location at home, we still need to be able to send the commands to the smart device. We setup an experiment to evaluate the performance of CTC at different distances in a residential apartment shown in Fig 13. We setup five TelosB motes as receivers at 7ft increment in distance from the AP being used as the CTC transmitter. The AP is a commercial Buffalo router running OpenWRT. The transmission power for the WiFi

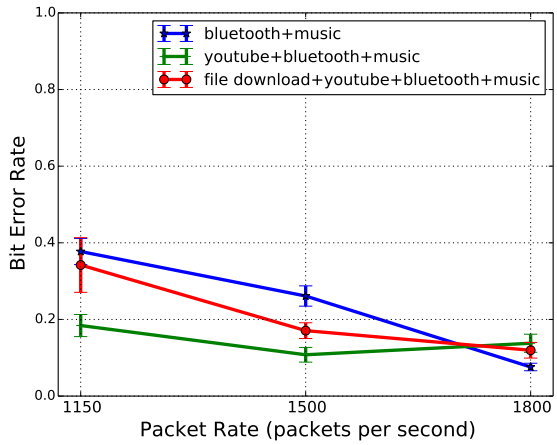

Fig. 18. BER vs UDP data rate with different WiFi traffic scenarios.

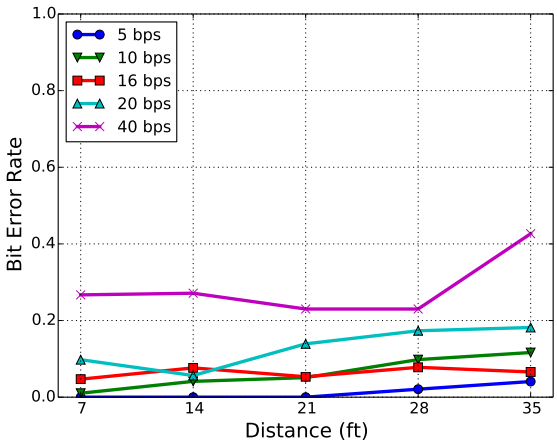

Fig. 19. BER vs. distance with different CTC data rates.

access point was configured as 17dBm. Fig 19 shows the result. In this residential apartment, there are multiple WiFi devices including cell phones, laptops, wireless printer, wireless access points and other reachable access points nearby. The motes sampled RSSI at 4 kHz. We run the experiment with UDP packets generated at 1000/s. We plot the BER vs distance for different CTC data rate in Fig 19

We find that the CTC data rate of up to 16 bps can achieve BER less than 10%. Further, the BER is stable within the 35ft X 35ft physical range, which is sufficient for typical CTC usage scenario at an apartment. This CTC data rate of 16 bps is sufficient to send commands to smart devices at homes. With lower CTC data rates, the BER can be close to 0 at moderate distances.

V. Discussions

In this section, we discuss different aspects of CTC design and performance issues.

1. ***Unidirectionality***. Our current implementation of CTC is unidirectional. Only the WiFi devices can send information to the 15.4 devices. Implementation of CTC in this direction is easier than CTC from 15.4 devices to WiFi. We can easily get WiFi transmission to saturate the 15.4 receiver and hence distinguish the times of transmission from times with no transmission. The main challenge in getting CTC to work from 15.4 to WiFi is getting the WiFi radio to detect 15.4 transmissions, which do not have a lot of power, from other transmissions in the crowded 2.4 GHz range. The implementation may be feasible in a commercial WiFi NICs but may require changes to the firmware for low-level access to the device for spectral scans.

2. ***Energy Consumption***. Our implementation requires the 15.4 devices to turn on theirs radios to listen to the ambient wireless signals, thus greatly weakening the design goal for low power wireless sensor networks. However, we can reduce the power consumption by coordinating the radio on and off times with the WiFi devices. Many smart gadgets in smart homes, however may be powered. If the 802.15.4 devices are powered, leaving the 15.4 radio on all the time may be acceptable.

3. ***Data Rate***. With the proposed techniques, we have achieved a data rate of 16 bps. Theoretically, we can achieve a data rate of 3 kbps with the maximum packet rate transmission on the WiFi devices, however that will require RSSI sampling and decoding at much faster rate on the motes. Operating at such high rates may also cause the BER to increase. Besides the challenge in high-speed RSSI sampling, symbol alignment also becomes challenging in a WiFi network with other traffic. Furthermore, the traffic generation must be real-time to ensure that the symbol duration is accurate. Otherwise, the decoding signal will not be synchronized with the encoded signal. Fortunately, even a low data rate CTC is useful for device configuration and commands and we expect CTC to be useful for those applications.

VI. Conclusions

We designed and implemented a WiFi to 15.4 communication system that utilizes crosstalk between the channels to deliver useful information between the devices. The proposed technique allows WiFi devices to directly communicate with 802.15.4 devices without any physical gateway. We provide a detailed description of the modulation and demodulation schemes and their evaluations in controlled and uncontrolled environment. The results show our proposed system can provide a reliable wireless communication to interconnect WiFi devices with IEEE 802.15.4 sensor nodes with an achieved data rate of 2 bytes per second with less than 10% bit error rate.

Acknowledgment

We thank Xiyao Xin and Prof. Ji Chen from the Department of Electrical and Computer Engineering at the University of Houston for providing the anechoice chamber for our experiments and feedback about experiments in the chamber. We thank Hessam Mohammadmoradi for inspiring discussions about the algorithm in this paper. This work was partially supported by the National Science Foundation under grant no. IIS-1111507.

REFERENCES

[1] V. L. Erickson and A. E. Cerpa, "Thermovote: Participatory sensing for efficient building hvac conditioning," in *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, ser. BuildSys '12. New York, NY, USA: ACM, 2012, pp. 9–16.

[2] T. W. Hnat, E. Griffiths, R. Dawson, and K. Whitehouse, "Doorjamb: Unobtrusive room-level tracking of people in homes using doorway sensors," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '12. New York, NY, USA: ACM, 2012, pp. 309–322.

[3] J. Ranjan, E. Griffiths, and K. Whitehouse, "Discerning electrical and water usage by individuals in homes," in *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, ser. BuildSys '14. New York, NY, USA: ACM, 2014, pp. 20–29.

[4] A. Frye, M. Goraczko, J. Liu, A. Prodhan, and K. Whitehouse, "Circulo: Saving energy with just-in-time hot water recirculation," in *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, ser. BuildSys'13. New York, NY, USA: ACM, 2013, pp. 16:1–16:8.

[5] G. Gao and K. Whitehouse, "The self-programming thermostat: Optimizing setback schedules based on home occupancy patterns," in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, ser. BuildSys '09. New York, NY, USA: ACM, 2009, pp. 67–72.

[6] Z. Shelby and C. Bormann, *6LoWPAN: the wireless embedded internet*. John Wiley & Sons, 2011, vol. 43.

[7] "6lowpan," https://tools.ietf.org/html/rfc4944, Sep. 2007.

[8] J. Hui, D. Culler, and S. Chakrabarti, "6lowpan: Incorporating ieee 802.15. 4 into the ip architecture," *IPSO Alliance White Paper*, vol. 3, 2009.

[9] J.-P. Vasseur and A. Dunkels, *Interconnecting smart objects with ip: The next internet*. Morgan Kaufmann, 2010.

[10] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '05. Piscataway, NJ, USA: IEEE Press, 2005.

[11] K. Chebrolu and A. Dhekne, "Esense: Communication through energy sensing," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '09. New York, NY, USA: ACM, 2009, pp. 85–96. [Online]. Available: http://doi.acm.org/10.1145/1614320.1614330

[12] S. Gollakota, F. Adib, D. Katabi, and S. Seshan, "Clearing the rf smog: making 802.11 n robust to cross-technology interference," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 170–181, 2011.

[13] A. Hithnawi, H. Shafagh, and S. Duquennoy, "Tiim: Technology-independent interference mitigation for low-power wireless networks," in *To appear in the Proceedings of the 14th ACM International Conference on Information Processing in Sensor Networks (IPSN '15). Seattle, WA, USA*, Apr. 2015.

[14] V. Iyer, F. Hermans, and T. Voigt, "Detecting and avoiding multiple sources of interference in the 2.4 ghz spectrum," in *Wireless Sensor Networks*, ser. Lecture Notes in Computer Science, T. Abdelzaher, N. Pereira, and E. Tovar, Eds. Springer International Publishing, 2015, vol. 8965, pp. 35–51. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-15582-1_3

[15] F. Hermans, O. Rensfelt, T. Voigt, E. Ngai, L.-Å. Norden, and P. Gunningberg, "Sonic: classifying interference in 802.15. 4 sensor networks," in *Proceedings of the 12th international conference on Information processing in sensor networks*. ACM, 2013, pp. 55–66.

[16] J.-H. Hauer, A. Willig, and A. Wolisz, "Mitigating the effects of rf interference through rssi-based error recovery," in *Wireless Sensor Networks*. Springer, 2010, pp. 224–239.

[17] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis, "Surviving wi-fi interference in low power zigbee networks," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2010, pp. 309–322.

[18] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma, "Zifi: wireless lan discovery via zigbee interference signatures," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 49–60.

[19] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, "An empirical study of low-power wireless," *ACM Trans. Sen. Netw.*, vol. 6, no. 2, pp. 16:1–16:49, Mar. 2010.

[20] S. Yin, O. Gnawali, P. Sommer, and B. Kusy, "Multi Channel Performance of Dual Band Low Power Wireless Network," in *Proceedings of the 11th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2014)*, October 2014.

[21] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: wireless communication out of thin air," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 39–50.

[22] P. Zhang, P. Hu, V. Pasikanti, and D. Ganesan, "Ekhonet: High speed ultra low-power backscatter for next generation sensors," in *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '14. New York, NY, USA: ACM, 2014, pp. 557–568.

[23] D. Giustiniano, N. O. Tippenhauer, and S. Mangold, "Low-complexity visible light networking with led-to-led communication," in *Wireless Days (WD), 2012 IFIP*. IEEE, 2012, pp. 1–8.

[24] Q. Wang, D. Giustiniano, and D. Puccinelli, "Openvlc: Software-defined visible light embedded networks," in *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems*. ACM, 2014, pp. 15–20.

[25] N. Rajagopal, P. Lazik, and A. Rowe, "Visual light landmarks for mobile devices," in *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '14. Piscataway, NJ, USA: IEEE Press, 2014, pp. 249–260. [Online]. Available: http://dl.acm.org/citation.cfm?id=2602339.2602367

[26] K. S. Lee, H. Wang, and H. Weatherspoon, "Phy covert channels: Can you see the idles?" in *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. Seattle, WA: USENIX Association, Apr. 2014, pp. 173–185. [Online]. Available: https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/lee

[27] "Wi-spy datasheet," http://files.metageek.net/marketing/data-sheets/MetaGeek_Wi-Spy-Chanalyzer_DataSheet.pdf.

[28] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 95–107.