StripComm: Interference-Resilient Cross-Technology Communication in Coexisting Environments

Xiaolong Zheng, Yuan He, Xiuzhen Guo
School of Software and TNLIST, Tsinghua University, Beijing, P. R. China
{zhengxiaolong, heyuan}@mail.tsinghua.edu.cn, guoxz16@mails.tsinghua.edu.cn

***Abstract*—Cross-Technology Communication (CTC) is an emerging technique to enable the direct communication among different wireless technologies. A main category of the existing proposals on CTC propose to modulate packets at the sender side, and demodulate them into 1 and 0 bits at the receiver side. The performance of those proposals is likely to degrade in a densely coexisting environment. Solely judged according to the received signal strength, a symbol 0 that is modulated as packet absence is generally indistinguishable from dynamic interference. In this paper, we propose *StripComm*, interference-resilient CTC in coexisting environments. A sender in StripComm adopts an interference-resilient coding scheme that contains both presence and absence of packets in one symbol. The receiver strips the interference from the interested signal by exploiting the self-similarity of StripComm signals. We prototype *StripComm* with commercial WiFi, ZigBee devices and a software radio platform. The theoretical and experimental evaluation demonstrate that *StripComm* offers a data rate up to 1.1K bps with a SER (Symbol Error Rate) lower than 0.01 and a data rate of 0.89K bps even against strong interference.**

I. Introduction

The proliferation of Internet of Things (IoT) applications calls for ubiquitous connections among various devices. The unprecedented prosperity of IoT brings explosive development of wireless devices as well as the rich diversity of wireless technologies. Interconnecting the heterogeneous devices that operate in the shared medium is a crucial but challenging task, because different technologies are essentially incompatible with each other [1]–[3]. The traditional way is to deploy dedicated gateways with multiple radios to relay data packets. Such gateway-based methods have extra hardware cost, increased traffic overhead and end-to-end delay induced by the relays.

Cross-Technology Communication (CTC) is an emerging technique to enable direct data exchange among heterogeneous wireless technologies. The core idea is to form a mutually accessible side channel as the information carrier. A widely recognized carrier is energy emission, which can be modulated while transmitting packets and detectable by any device that operates at the same bands. By manipulating the timing, the frequency, and/or the amplitude of packets, data can be modulated on the energy channel. For example, FreeBee [4] shifts the transmission time of periodical beacons to encode data in the temporal dimension. The data rate of FreeBee, however, is bounded by the low beacon rate (one beacon/100*ms* for default commercial WiFi devices). Increased data rate is desired to facilitate the cooperations of heterogeneous devices for smarter context sensing [5]–[8] and data analysis [9], [10]. The work in

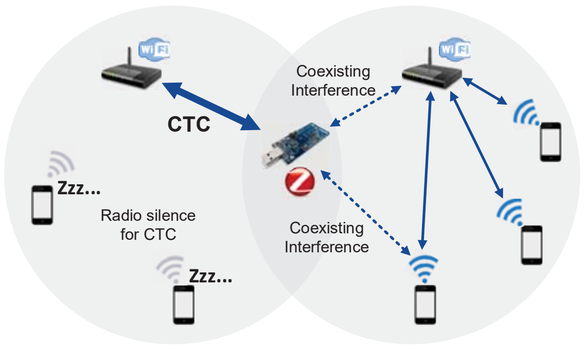


Fig. 1. CTC in coexisting environments

[11] controls the amplitude and the frequency of transmissions to mimic a discrete amplitude and frequency shift keying (DAFSK) converter. It enables the communication from BLE to WiFi. The works in [12] and [13] control the presence and absence of packets to encode bits 1 and 0 in the amplitude dimension. Multiple amplitudes are utilized in WiZig [14] to encode more bits simultaneously.

Though the existing works are proposed for CTC among coexisting devices, most of them don't sufficiently address the challenges induced by coexisting interference. The work in [11] allows concurrent transmissions from controlled WiFi devices rather than uncontrolled interferers. Only a few existing proposal take communication reliability into account, while they merely endure noise by repeating transmissions [4] or adding redundant bits to encode one symbol [14]. The throughput of CTC are sacrificed in those cases. On the other hand, the explosive increase of deployed wireless IoT devices leads to the increased chance of collisions between CTC and interferers. Fig. 1 shows an example of CTC in coexisting environments. Although CTC can be free from the interference in the senders communication range by using RTS/CTS to reserve the channel, it is still easy for other ambient devices to introduce serious performance degradation of CTC. According to our experimental study, the SER (Symbol Error Rate) of amplitude-modulation based CTC can be increased for an order of magnitude in a typical coexisting environment. Considering the practice of IoT applications, how to make CTC resilient to interference is still an open problem.

In this paper, we propose *StripComm*, a novel interference-resilient CTC technique tailored to the coexisting environments. Through both theoretical and practical studies, we find the root cause of performance degradation of CTC is that traffic from uncontrolled coexisting devices easily causes false presence of CTC packets, confusing the coded pattern of CTC. To reduce the impacts of false presence, *StripComm* modulates both presence and absence of packets in a single symbol. To further improve the performance, we devise a novel interference-aware decoding mechanism that strips out the interference from the interested CTC signal by exploiting the self-similarity of *StripComm* symbols. The main contributions of this work are summarized as follows.

- We propose *StripComm*, a novel CTC technique interconnecting WiFi and ZigBee devices in coexisting environments. We design a new interference-resilient modulation mechanism that encodes symbols by the changes of packet presence and absence to avoid the fallibility of the single state.
- We devise an interference-aware decoding mechanism that strips out the interference based on the distinguishable RSS patterns caused by the self-similarity of *StripComm* signals. Demodulating the recovered signals with interference cancelation increases the reliability of *StripComm*.
- We implement a prototype of *StripComm* with commercial devices and a software radio platform. We evaluate the performance of *StripComm* under various experimental settings. The throughput of *StripComm* is 1.1K bps with SER lower than 0.01 in a real office environment, and still 0.89K bps even under strong interference.

The rest of this paper is organized as follows. Section II presents the related work. In Section III, we observe the CTC performance in coexisting environments and theoretically analyze the root cause of performance degradation. We introduce the design details of *StripComm* in Section IV. We evaluate the performance of *StripComm* in Section V and conclude our work in Section VI.

II. Related Work

Interconnecting all the smart things is an inevitable trend of IoT. Cross-Technology Communication (CTC) is a promising technique to enable direct data exchanges among heterogeneous devices. FreeBee [4] is a representative work that modulate data in time dimension by shifting the transmission timings of beacons. The data rate is therefore bounded by the low beacon rates. ESense [12] is the first work realizing energy-based communication from WiFi to ZigBee devices. It encodes the alphabet into different packet lengths. HoWiEs [15] improves ESense by using combinations of WiFi packets to convey data. GSense [16] uses a customized packet preamble and modulate symbols by the gaps between energy pulses.

Recently, the authors in [13] propose to use the amplitude modulation and encode symbol 1/0 by the presence/absence of packets. WiZig [14] improves [13] by using multiple amplitudes to encode more symbols simultaneously. C-Morse [17] controls the packet presence with Morse Coding to convey the information. B2W2 [11] modulates symbols in both amplitude and frequency to mimic the DAFSK for communication from BLE to WiFi devices.

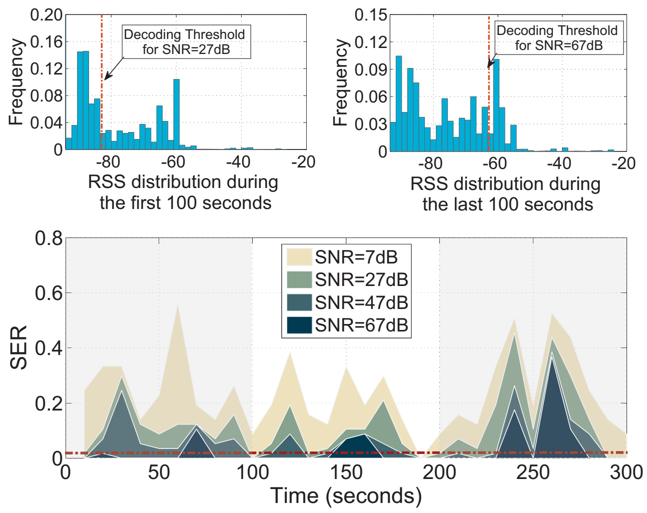


Fig. 2. SER of WiZig and the distributions of interference RSS during our experiments in an apartment, a typical coexisting environment.

WEBee [18] is a more recent work that directly emulates the ZigBee signal by WiFi packets in the physical layer. However, emulating the ZigBee signal with WiFi payload is suspected to prefer a good signal-to-interference-ratio (SINR). Otherwise, coexisting interference will corrupt the coded payloads.

StripComm is different with the aforementioned related works from two aspects. First, *StripComm* is an interference-resilient CTC tailored to the coexisting environments while existing works usually passively endure the coexisting interference as noise. Second, *StripComm* integrates a novel interference-aware demodulation mechanism that strips out the influence of interference. Interference classification has been studied in [19]–[22]. Different from these works, *StripComm* leverages the self-similarity of *StripComm* signals to cancel the interference from CTC signals.

III. Preliminary Study

We conduct experiments in a typical coexisting environment to investigate the performance of amplitude modulation based CTC (AM CTC) in coexisting environments. AM CTC uses packet presence and absence to encode symbol 1 and 0 [13] [14]. During the experiments, AM CTC transmits '10' repeatedly for five minutes and the symbol error rate (SER) is recorded every 10 seconds. The window length of one symbol is 5*ms*. The WiFi sender operates on WiFi channel 1 and ZigBee receiver operates on ZigBee channel 13. The receiver keeps radio on without the duty-cycle operation [23]. We don't add any intentional interference and AM CTC only experiences the uncontrolled ambient interference. We control the transmission power to obtain the received signal strength (RSS) of -90dB, -70dB, -50dB and -30dB at the ZigBee receiver, i.e., the SNR of 7dB, 27dB, 47dB and 67dB when the noise is -97dB.

The results are shown in Fig. 2. When SNR is 7dB, SER is 0.24 in average and larger than 0.5 during $[50s, 60s]$. Even when we configure the RSS as -30dB (SNR $=67dB$), the SER is still larger than 0.35 during $[250s, 260s]$. We plot the RSS distributions during the first and the last 100 seconds in Fig. 2. We can find that during the first 100 seconds, more than 40% RSS are smaller than -84dB. Given the decoding threshold as half of the signal strength, using a RSS of -70dB for CTC signal greatly reduces the SER. However, during the last 100 seconds, we can find around 20% RSS are larger than -62dB. Hence, even increasing RSS of CTC signal to -30dB, the coexisting interference can easily increase the SER.

The reason behind the results is existing AM CTC methods just treat the coexisting interference as noise and rely on a good SINR to avoid the interference. However, a high SINR is not easy to obtain in practice. For example, in our experiments, to conquer the interference, the signal RSS of AM CTC needs to be even larger than -13dB. To solve this problem, existing works repeat transmissions [4] or add redundant bits in one symbol [14]. But such enduring methods sacrifice the throughput.

IV. Design

Fig. 3 shows the architecture of *StripComm*. A logical channel of *StripComm* is built on top of the existing technologies' physical channel, without any modified or dedicated hardware. According to the preliminary studies, existing AM CTC suffers from coexisting interference because of the false packet presences during symbol 0 when packet absence should be detected. Therefore, we propose *iCoder*, an interference-resilient coding mechanism (Section IV-A). Instead of solely using packet presence or absence to encode a symbol, *iCoder* leverages both the presence or absence of packets for one symbol according to Manchester Coding. By introducing the high RSS bits (packet presence), *iCoder* is resilient to the interference happening during the packet presence. But using less low bits (packet absence) also increases the chance of being affected by the shorter interference. To settle this issue, we propose *iDecoder*, an interference-aware decoding mechanism (Section IV-B) that strips the interference from the interested CTC signals by exploiting the self-similarity of *StripComm* signals.

A. Interference-Resilient Coding

iCoder uses both packet presence and absence to encode a symbol according to Manchester Coding, augmenting the resilience of symbol 0 to interference. As shown in Fig. 4, *iCoder* divides a symbol window into two halves and controls the packet presence in the first half or the latter one to represent symbol '1' or '0'. The receiver then detects the rising and failing edges to decode the corresponding symbols.

By introducing high bits (packet presence) into symbol 0, *iCoder* is resilient to the interference that happens during the high bits and still effective even when parts of the low bits are flipped by the interference. As illustrated in Fig. 5, when an interference with the length of L happens, no matter

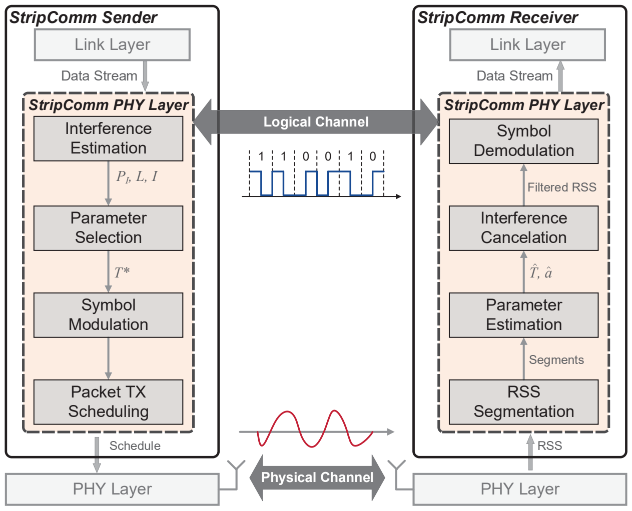


Fig. 3. The architecture of *StripComm*

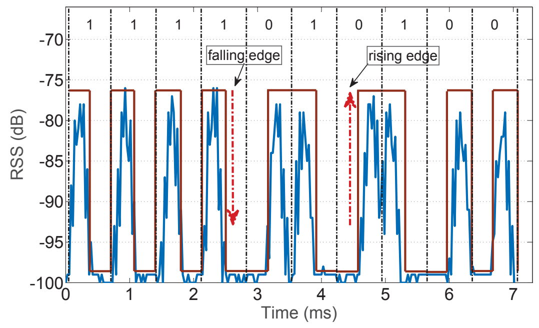


Fig. 4. *StripComm* encodes a symbol with both packet presence and absence

when the interference starts, AM CTC will incorrectly decode the second symbol as '1' as long as $L > m$, where m is the threshold of the high RSS bits to determine the packet presence. However, *iCoder* correctly decodes this symbol because only $\overline{L}$ low bits expose to the interference and the other $L - \overline{L}$ high bits are unlikely flipped.

Existing AM CTC methods increase the symbol window length (T) to resist the channel noise. Nevertheless, increasing T can bring more interfered bits. Suppose the traffic duty cycle of a coexisting device is P_I, the average interference strength and the average packet length are I and L, then K should not exceed L/P_I. Otherwise, more interfered bits will be introduced and the SER can be even higher. According to existing measurement studies [24], [25], the channel occupancy rate of coexisting interference can be high even when considering only one device, not mention the large number of coexisting devices in the IoT systems. With a bounded T, *iCoder* can still be resilient to the interference, making it possible to achieve a higher data rate.

1) PHY layer design on StripComm senders: A *StripComm* sender tries to provide the maximum throughput with a bounded SER. Note that most of the existing wireless technologies have the minimum inter-packet interval (*IPI*) requirement,

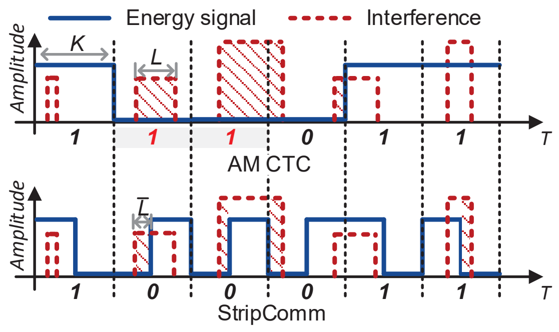


Fig. 5. *iCoder* is resilient to interference

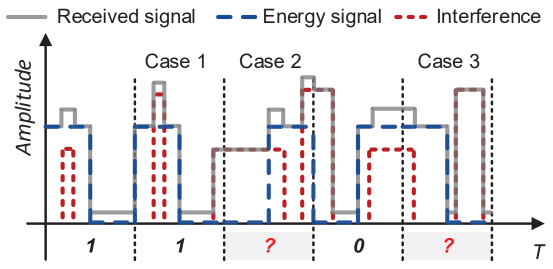


Fig. 6. Corrupted RSS sequence of *StripComm* under coexisting interference

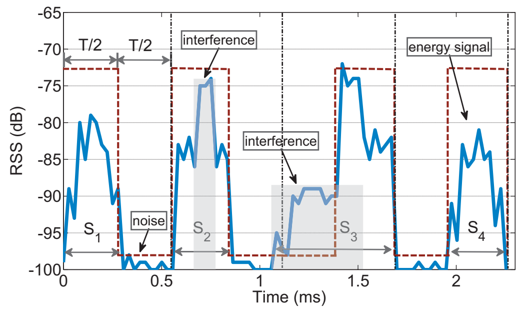


Fig. 7. Case 1 & Case 2: Overlapped interference corrupts symbols

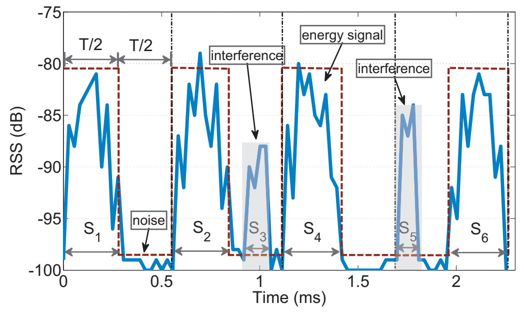


Fig. 8. Case 3: Inserted interference corrupts symbols

such as $28\mu s$ (DIFS) for WiFi and $192\mu s$ for ZigBee. Denote the sampling period as T_s. Then the minimum length of a symbol window should be $2 \cdot IPI/T_s$. Besides, the window length should be at least longer than the interference. Otherwise, every single bit will be heavily influenced by interference.

We should set the symbol window length, T^*, to optimize the throughput with a limited SER. Therefore, information about interference including P_I, I, and L are needed. Estimating interference information at the receiver is more accurate because of the spatial diversity. Therefore, a *StripComm* receiver estimates the interference. But the interference estimation component is done on the device with more powerful computational ability because calculating T^* has high computation cost for low-power devices. If the *StripComm* receiver is more powerful, it directly calculates T^* based on the estimated interference information and piggyback T^* in ACK messages to the sender. Otherwise, the *StripComm* receiver will piggyback the raw interference information in ACK messages and sends backs to the sender. Then the *StripComm* sender is responsible for calculating T^*.

Given the selected parameters, *StripComm* generates a packet transmission schedule for the underlying physical layer. *StripComm* will first try to generate a single packet with the on-air time equals to $T^*/2$ to avoid the blank space caused by *IPI*. If $T^*/2$ is larger than the maximum allowed on-air time, *StripComm* will schedule a bursty $T^*/2$ transmission with multiple packets to generate high bits in a symbol window.

B. Interference-Aware Decoding

iCoder is resilient to coexisting interference but not able to eliminate the interference. When interference occurs during the low bits, both symbol '1' and '0' can be incorrect. Dividing a symbol window into two halves also increases the chance of being influenced by shorter interference because less low bits are used in *iCoder*, compared to the traditional AM CTC methods. For example, the third and fifth symbols in Fig. 6 are corrupted because the interference occurs during low bits.

To tackle this problem, we devise a novel interference-aware decoding mechanism called *iDecoder*. *iDecoder* strips the interference from the RSS sequence and recovers the interested RSS sequence by exploiting the **self-similarity** of *StripComm* signal. The self-similarity of *StripComm* signals is reflected from the following three aspects.

- **Ratio similarity**: the numbers of high and low bits in each *iCoder* symbol are equivalent. Then in any $n * T$ time, the duty cycle of *StripComm* signal is 50%.
- **Time similarity**: the numbers of high bits in different symbols are equivalent during one packet transmission because *StripComm* only adjusts the parameters when preparing for the next new packet but not inside a packet.
- **Amplitude similarity**: the signal strength from the same sender is usually stable with limited fluctuations during a short period such as tens of milliseconds.

1) Coexisting interference corrupts symbols: Based on our studies, most of the interfered segments can be divided into three cases, as illustrated by Case 1, 2 and 3 in Fig. 6. In Case 1 and 2, coexisting interference coming from hidden terminals overlaps with *StripComm* signals. In Case 3, the interference occurs during the low bits because the coexisting devices are trying to leverage the white space between the *StripComm* transmissions. In Fig. 7 and Fig. 8, we present the

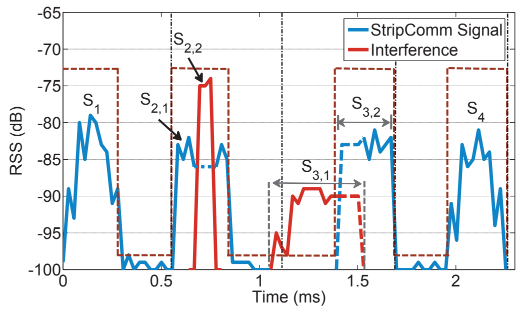


Fig. 9. Separated segments of the RSSI trace in Fig. 7

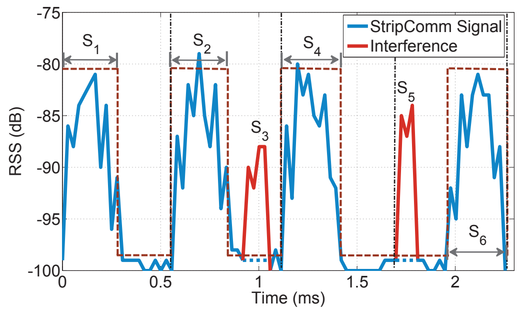


Fig. 10. Separated segments of the RSSI trace in Fig. 8

TABLE I
SIMILARITY FEATURES USED BY INTERFERENCE CANCELATION

Feature	Description	Self-similarity
DC_i	Duty cycle of the window containing S_i	Ratio similarity
T_i	Segment length of S_i	Time similarity
ISI_i	Inter-segment interval between S_i and the last segment marked as signal	
a_i	Average amplitude of segment S_i	Amplitude similarity
δ_i^a	Variance of the amplitudes ofS_i	

real data traces containing Case 1, 2 and Case 3. We can find that interference in Case 1 corrupts the amplitude similarity of the interested signals but not the time and ratio similarity. The interference in Case 2 and Case 3 is able to corrupt all the three similarities.

To eliminate the impacts of interference and recovery the corrupted segments, *iDecoder* leverages the distinguishable RSS patterns caused by the self-similarity of *StripComm* signal. But during a single symbol window, the ratio similarity is fallible and neither the time nor amplitude similarity is available. For example, in Case 2, we are not even sure whether there is a *StripComm* symbol because the duty cycle is much larger than 50%. Therefore, instead of demodulating the symbols one by one, *iDecoder* demodulates D symbols in a batch with more accurate extracted features.

2) RSS segmentation: As shown in Fig. 3, a *StripComm* receiver listens to the channel and collects the RSS samples. The collected RSS sequence is segmented to extract the high RSS segments that are possibly related to the high bits of *StripComm* symbols. It is known that the start of a signal causes a rising edge and the stop of a signal leads to a falling edge in the RSS sequence. Hence, we adopt a simple yet effective change point detection method, CUSUM, to check the start and stop of a signal. Then the RSS sub-sequence during the detected start and stop is regarded as a segment. A segment S_i has three attributes: the start time t_i^{start}, the stop time t_i^{stop}, and the average RSS a_i. The set of segments are sorted in start time order, $\mathbf{S} = \{S_1, S_2, ..., S_N\}$.

3) Parameter estimation: Existing works rely on pre-set parameters or exchange the parameters by dedicated control messages. *StripComm* can also exchange these parameters in advance, omitting the estimation process. Due to the self-similarity, an alternative way of *StripComm* is estimating parameters online, eliminating the communication cost of control messages. With a bunch of segments, *iDecoder* can estimate the symbol window length T and the amplitude a. *iDecoder* checks each segment S_i to see any space between the adjacent segments $((t_i^{start} - t_{i-1}^{stop})$ and $(t_{i+1}^{start} - t_i^{stop}))$ has the same length as its own length with an allowed error, ε_T. We mark the found segment as a possible uncorrupted symbol and record the corresponding $2 * T_i$ and a_i in $\mathbf{T}$ and $\mathbf{a}$. Then we select the mode of $\mathbf{T}$ as the estimated window length of a symbol, $\hat{T}$, and the average RSS of segments with length $\hat{T}/2$ as the estimated amplitude $\hat{a}$.

4) Interference cancelation: Given the parameters $\hat{T}$ and $\hat{a}$, *iDecoder* deals with the three types of corruptions by the interference cancelation component. First of all, *iDecoder* separates the interference segments from the raw segments. We observe that the start of an overlapped signal also causes a rising edge and the stop of either signal causes a falling edge, as shown in Fig. 7. Note that the strength of two interfered signal can be enhanced (constructive interference) or weaken (destructive interference). But the destructive interference requires the devices send the same data and the signals arrive at the receiver at the same time. In coexisting environments, the signals from two uncontrolled devices generating different data are unlikely to have destructive interference. Therefore, the rising and falling edges are observed.

We apply the CUSUM-based segmentation again on the interference-suspected segments to detect the hidden segments. If a rising edge is found, a possible overlapped segment is found and we record the start position in a record stack. If a falling edge is found and more than one starts are in the stack, then we need to judge which start the detected stop matches. We compare the average RSS of subsegment before the latest start and the average RSS of subsegment after the currently detected stop. If they are comparable, we regard they are from the same device based on amplitude similarity and match the detected stop with the latest start, such as $S_{2,1}$ and $S_{2,2}$ in Fig. 9. Otherwise, we match the detected stop with the former start, such as $S_{3,1}$ and $S_{3,2}$ in Fig. 9. The newly separated segments are added into $\mathbf{S}$ and the originally overlapped segment is removed from $\mathbf{S}$.

iDecoder then eliminates the interference segments in $\mathbf{S}$ according to the features of the self-similarity of *StripComm*

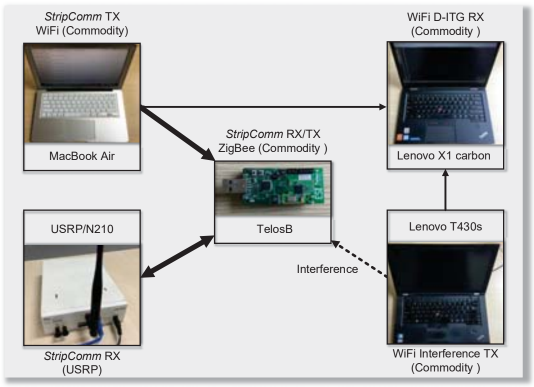


Fig. 11. Experiment setup

signals. The features we used are listed in Table I. For the valid interested signal segments in a demodulation window, the features will meets: (1) the ratio similarity, $DC_i = 50\%$, (2) the time similarity, $T_i = \hat{T}/2 + \varepsilon_T$, $ISI_i = T_i^{start} - T_j^{stop} = 0$ *or* $\hat{T} \pm 2\varepsilon_T$, and (3) the amplitude similarity, $a_i = \hat{a} \pm \varepsilon_a$, $\delta_i^a = \delta_j^a \pm \varepsilon_{va}$, where S_j is the last segment that is marked as an interested signal. By checking the features, interference segments and interested signal segments can be separated. For example, for case 1 and 2, in Fig. 9, $S_{2,2}$ and $S_{3,1}$ are removed because they violate the time and amplitude similarity. For Case 3, S_3 is removed due to the violation of all the three similarities. Even though S_5 has a similar amplitude with the interested signal, it violates the time similarity and therefore will be removed as well. By the self-similarity, *iDecoder* strips the interference from the received RSS sequence and recoveries the interested RSS sequence.

5) Symbol demodulation: With the set of segments after interference cancelation, $\mathbf{S}^*$, *iDecoder* regenerates the RSS sequence and follows the standard demodulation process to decode each symbol. A practical communication system usually has a preamble for each packet. In *StripComm*, we define the preamble as eight '1's. The default decoding window length D is also set to 8 for detecting the preamble with a low latency. The *StripComm* receiver listens to the channel and seeks for the preamble. Once detecting the preamble, the *StripComm* receiver uses the estimated parameters, $\hat{T}$ and $\hat{a}$, for following symbols demodulation to avoid redundant estimation process.

V. Evaluation

A. Experiment setup

The platforms used in our evaluation are shown in Fig. 11. *StripComm* does not rely on any dedicated hardware. The feasibility of implementing *StripComm* on commercial devices because (1) controlling the packet transmissions is possible on both WiFi devices and TelosB platform, (2) the only needed information at the receiver is RSS which is a common indicator provided by wireless devices.

ZigBee: We use TelosB, a commercial ZigBee platform, to implement the *StripComm* ZigBee sender and receiver. The ZigBee sender controls the payload length and the transmission timings between successive packets to generate *StripComm* symbols. In our current implementation, the symbol window $T = 8ms$ for the ZigBee sender. The RSS sampling frequency on TelosB is 36KHz. The communication channel of ZigBee devices is set to channel 23 unless otherwise specified.

WiFi: We use commercial computers with IEEE 802.11 b/g/n radios operating in 2.4GHz band as *StripComm* WiFi senders. The modulated traffic is generated by D-ITG [26], a traffic generator supports bursty transmissions with user-defined on/off time. We use a software defined radio platform, USRP/N210, to implement a *StripComm* WiFi receiver. Because commercial WiFi devices usually extract RSS from the received WiFi packet, providing a limited RSS sampling rate. Hence, we use USRP/N210 to collect the RSS continuously with a frequency of 36KHz. The communication channel of WiFi devices is set to channel 11 unless otherwise specified.

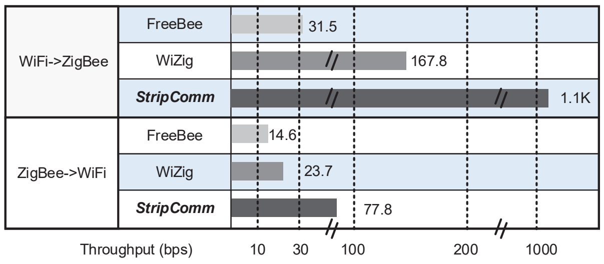


Fig. 12. Comparison with the state of the arts.

B. Overall Performance Comparison

We present the overall performance of *StripComm*, compared with the state-of-the-art works. To evaluate the communication from WiFi to ZigBee, we deploy a laptop as the *StripComm* sender and four TelosB as *StripComm* receivers at different locations in our lab, with the distance from 1 to 9 meters. For the communication from ZigBee to WiFi, we deploy a TelosB node as the *StripComm* sender, two meters away from the USRP/N210 that acts as the receiver. The window length of a symbol is set to $T = 0.896ms$ ($K = 32$ bits to encode a symbol). For comparison, we also implement WiZig [14] based on its design principle. For FreeBee, we cite its reported performance in [4] for simplicity. The experiments are done during working hours with about a dozen of WiFi APs and several Bluetooth headsets operating around.

The comparison results are shown in Fig. 12. For communication from WiFi to ZigBee, with a SER lower than 0.01, *StripComm* can achieve a throughput of 1.1K bps in average, which is 6.5× and 34.9× higher than WiZig and FreeBee, respectively. From ZigBee to WiFi, *StripComm* achieves a throughput of 77.8 bps, 3.3× and 5.3× higher than WiZig and FreeBee, respectively. The SER of *StripComm* is 0.12 and the SER of WiZig is 0.88. The asymmetric performance is because of the asymmetric transmitting power and bandwidth between WiFi and ZigBee. The low transmitting power of ZigBee on the narrow ZigBee bandwidth can only affects a limited part of the wide WiFi bandwidth, resulting in limited influence of

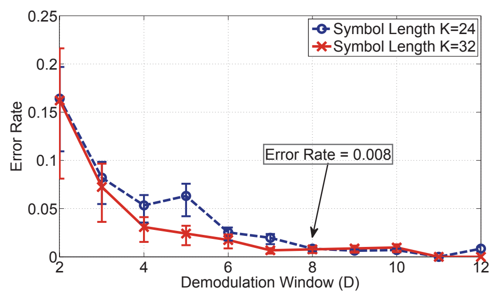


Fig. 13. Estimation error rate of the symbol window length K

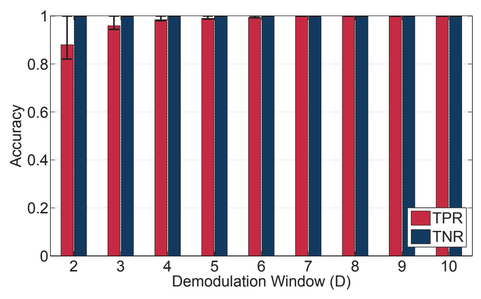


Fig. 15. Accuracy of interference cancelation in an office environment

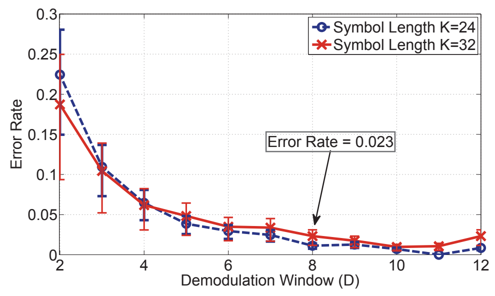


Fig. 14. Estimation error rate of the symbol amplitude a

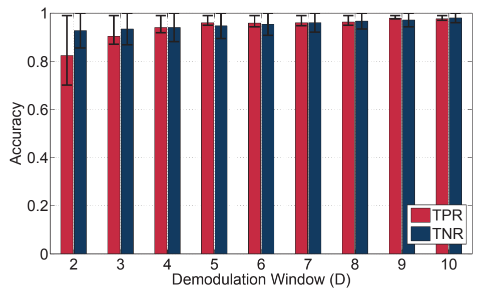


Fig. 16. Accuracy of interference cancelation with heavy interference

ZigBee transmission on RSS. Interference is therefore easy to corrupt the bits modulated by amplitude. But *StripComm* is still effective to improve the performance.

C. Accuracy of Parameter Estimation

StripComm can reduce control overhead by online parameter estimation instead of exchanging parameters in advance. The estimation relies on the distinguished RSS pattern caused by self-similarity of *StripComm* signal. The longer the observation time is, the clearer pattern is expected. We vary the demodulation window length D from 2 to 12 and use two adjacent symbol window lengths, $K = 24$ and $K = 32$, to investigate the estimation accuracy of symbol window length K and the amplitude a. The results are shown in Fig. 13 and Fig. 14. As expected, with the increase of D, error rates of both T and a decrease. When $D = 8$, the error rate of T and a are 0.008 and 0.023, which is good enough for following processes of *StripComm*. This is also the reason that we define the preamble length of *StripComm* as 8.

D. Accuracy of Interference Cancelation

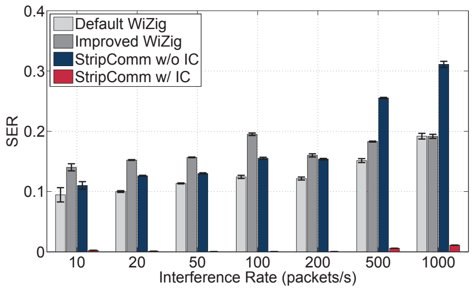
Interference cancelation (IC) is a core build block of *StripComm*. We evaluate the accuracy of IC in the office with only uncontrolled interference. To study the robustness of IC in heavy interference environments, we add a controlled interference TX laptop into the office, transmitting 500 packets per second. We repeat the experiment ten times. The experiment results are shown in Fig. 15 and Fig. 16. From the results, we can find both the True Positive Rate (TPR) and True Negative Rate (TNR) are nearly 1.0 when more than 7 symbols ($D \geq 7$) are batched to demodulate. After adding the controlled interference, the TPR and TNR drop to 0.979 and 0.972 if using the default demodulation window length $D = 8$. The accuracy degradation is very limited, demonstrating the robustness of IC. The unremoved interference can be left to be defected by the resilience of *StripComm* coding. The performance gain of IC will be further studied in Section V-F.

E. Impacts of Symbol Window Length

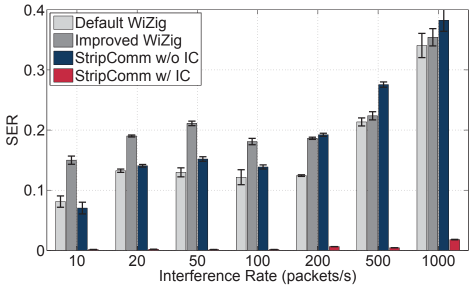
The symbol window length T decides the upper bound of throughput ($1/T$). A short symbol window increases the throughput but also increases the SER because the redundancy of encoded symbol is reduced. We conduct experiments in the office environment to study the performance of *StripComm* with four symbol window lengths from 0.672ms (24 bits/symbol) to 2.688ms (96 bits/symbol). The distance between the sender and receiver is 4 meters. For each setting, the sender repeatedly transmits symbol 0 or 1 for 10 seconds during one experiment. We repeat the experiment 10 times. The experiment results are summarized in Table II. When using a short window $T = 0.672ms$, the throughput is 1.45K bps but the SER is higher than 2%. When $T = 0.896ms$, the SER is reduced to 0.27% and the throughput is 1.11K bps. When further increasing T to 2.688ms, *StripComm* can be reliable but the corresponding throughput is only 0.37K bps.

F. Impacts of Interference Rate

We then evaluate *StripComm*'s performance in environments with controlled interference to demonstrate the resilience of *StripComm* to coexisting interference. We use D-ITG to generate UDP traffic between two laptops with different packet transmitting rate and packet size as interference. We use two

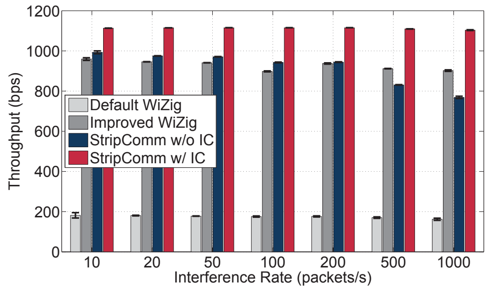


(a) Packet size=50bytes

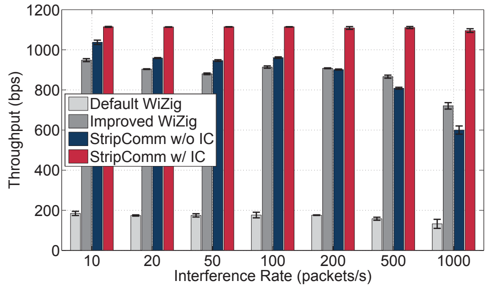


(b) Packet size=1000bytes

Fig. 17. SER vs. interference rates



(a) Packet size=50bytes



(b) Packet size=1000bytes

Fig. 18. Throughput vs. interference rates

TABLE II
SER, THROUGHPUT VS. LENGTH OF SYMBOL WINDOW

Symbol Length (T)	0.672ms	0.896ms	1.12ms	2.688ms
SER (%)	2.46	0.27	0.16	0
Throughput (bps)	1451.36	1113.09	891.43	372.02

interference packet sizes, 50$bytes$ and 1000$bytes$ to simulate the short and long lasting interference and vary the interference rate from 10 packets/s to 1000 packets/s. For each setting, the experiment is repeated 10 times. We compare WiZig with the default symbol window length (5ms/symbol) as default WiZig, WiZig with a symbol window length of 0.896ms as improved WiZig, *StripComm* without interference cancelation component as *StripComm* w/o IC, and *StripComm* with IC.

Fig. 17 and Fig. 18 show the SER and throughput of the four methods. With the increase of interference rate, all the methods suffer from performance degradation (SER increase and throughput degrease). When interference rate is 10 packets/s, *StripComm* offers a throughput of 1.115K bps and 1.114K bps with the SER lower than 0.01 under the short and long interference, respectively. The throughput is 616%, 112% higher than default WiZig and *StripComm* w/o IC. Though the improved WiZig offer a higher throughput than default WiZig, it has a much higher SER because using a short symbol window but without any defense to interference. When the interference rate increases to 1000 packets/s, the throughput of *StripComm* drops to 1.103K bps and 1.096K bps, decreasing by only 1.1% and 1.6% under the short and long interference, respectively. When increasing the rate from 10 to 1000, the throughput of default WiZig, improved WiZig and *StripComm* w/o IC decrease by 10.5%, 5.9% and 22.7% under the short interference, and decrease by 28.2%, 24.1% and 42.1% under the long interference. The stable performance of *StripComm* is because the interference cancelation effectively removes most of the interference and provides a low SER.

The SER of *StripComm* remains lower than 0.01 except when the rate is 1000 packets/s and the packet size is 1000 bytes (The SER is 0.018). To reduce the SER below 0.01, it is necessary to use a longer symbol window to improve the reliability. When $T = 0.56ms$, the SER is reduced to 0.0091 and the corresponding throughput is 0.885K bps. The SER of the other three methods are much higher than *StripComm*, ranging from 0.08 to 0.34 under different settings.

G. Impacts of Distance

We then evaluate *StripComm*'s performance when varying the distance between the sender and receiver from 1 to 9 meters. We conduct the experiments in an office and repeat the experiment 5 times at each location. Fig. 19 and Fig. 20 show the experiment results of *StripComm* from WiFi to ZigBee and from ZigBee to WiFi, respectively. The performance degrades with the increase of the distance for both communication directions, as expected. But *StripComm* from WiFi to ZigBee has a much smaller performance variation. The throughput only decreases by 0.6% and the SER only increases from 0.001 to 0.008. However, on the reverse communication direction, the performance degrades fast with the increase of distance. When the distance is 9 meters, the SER is nearly 1.0 and the throughput is only 2.08bps. The asymmetric performance is because the low transmitting power of ZigBee on the narrow ZigBee bandwidth can only affects a limited part of the wide WiFi bandwidth, resulting in limited influence of

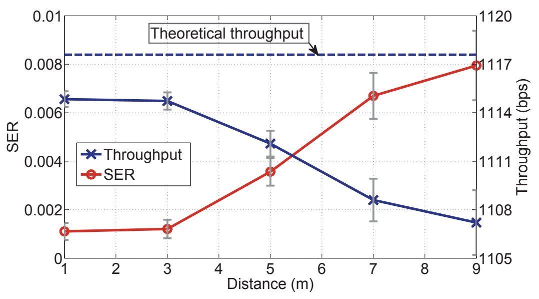


Fig. 19. SER and throughput vs. distance, from WiFi to ZigBee.

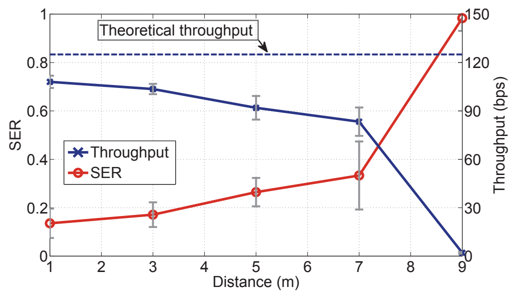


Fig. 20. SER and throughput vs. distance, from ZigBee to WiFi.

ZigBee transmission on RSS when distance is far. Instead of detecting the changes of RSS, measuring the changes of WiFi channel state information [27] may help to solve the asymmetric problem.

VI. Conclusion

In this work, we propose *StripComm*, an interference-resilient CTC tailored to the coexisting environments. By modulating a symbol with both packet presence and absence, *StripComm* is resilient to the interference that causes false p-resences. *StripComm* strips out the interference from interested signal by exploiting the self-similarity of *StripComm* signals. We prototype *StripComm* on commercial WiFi, ZigBee devices and a software defined radio platform. The evaluation results show *StripComm* provides a data rate up to 1.1K bps with a SER lower than 0.01 and a data rate of 0.89K bps even against strong interference.

Acknowledgment

This work was supported by National Key R&D Program of China 2017YFB1003000, National Basic Research Program (973 program) under Grant of 2014CB347800, The NSFC No. 61672320, and No. 61772306, National Natural Science Fund of China for Excellent Young Scientist No.61422207, and China Postdoctoral Science Foundation No. 2016M601034.

References

[1] T. Zachariah, N. Klugman, B. Campbell, J. Adkins, N. Jackson, and P. Dutta, "The internet of things has a gateway problem," in *Proceedings of ACM HotMobile*, 2015.

[2] C.-J. M. Liang, K. Chen, N. B. Priyantha, J. Liu, and F. Zhao, "Rushnet: practical traffic prioritization for saturated wireless sensor networks," in *Proceedings of ACM SenSys*, 2014.

[3] X. Zhang and K. G. Shin, "Enabling coexistence of heterogeneous wireless systems: Case for zigbee and wifi," in *Proceedings of ACM MobiHoc*, 2011.

[4] S. M. Kim and T. He, "Freebee: Cross-technology communication via free side-channel," in *Proceedings of ACM MobiCom*, 2015.

[5] J. Han, H. Ding, C. Qian, W. Xi, Z. Wang, Z. Jiang, L. Shangguan, and J. Zhao, "Cbid: A customer behavior identification system using passive tags," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2885–2898, 2016.

[6] Y. Jiang, Z. Li, and J. Wang, "Ptrack: Enhancing the applicability of pedestrian tracking with wearables," in *Proceedings of IEEE ICDCS*, 2017.

[7] H. Ding, J. Han, L. Shangguan, W. Xi, Z. Jiang, Z. Yang, Z. Zhou, P. Yang, and J. Zhao, "A platform for free-weight exercise monitoring with rfids," *IEEE Transactions on Mobile Computing*, vol. 16, no. 12, pp. 3279–3293, 2017.

[8] W. Gu, Z. Yang, L. Shangguan, W. Sun, K. Jin, and Y. Liu, "Intelligent sleep stage mining service with smartphones," in *Proceedings of ACM UbiComp*, 2014.

[9] Y. Tong, C. C. Cao, and L. Chen, "Tcs: efficient topic discovery over crowd-oriented service data," in *Proceedings of ACM SIGKDD*, 2014.

[10] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1334–1345, 2017.

[11] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu, "B2w2: N-way concurrent communication for iot devices," in *Proceedings of ACM SenSys*, 2016.

[12] K. Chebrolu and A. Dhekne, "Esense: Communication through energy sensing," in *Proceedings of ACM MobiCom*, 2009.

[13] S. Yin, Q. Li, and O. Gnawali, "Interconnecting wifi devices with ieee 802.15. 4 devices without using a gateway," in *Proceedings of IEEE DCOSS*, 2015.

[14] X. Guo, X. Zheng, and Y. He, "Wizig: Cross-technology energy communication over a noisy channel," in *Proceedings of IEEE INFOCOM*, 2017.

[15] Y. Zhang and Q. Li, "Howies: A holistic approach to zigbee assisted wifi energy savings in mobile devices," in *Proceedings of IEEE INFOCOM*, 2013.

[16] X. Zhang and K. G. Shin, "Gap sense: Lightweight coordination of heterogeneous wireless devices," in *Proceedings of IEEE INFOCOM*, 2013.

[17] Z. Yin, W. Jiang, S. M. Kim, and T. He, "C-morse: Cross-technology communication with transparent morse coding," in *Proceedings of IEEE INFOCOM*, 2017.

[18] Z. Li and T. He, "Webee: Physical-layer cross-technology communication via emulation," in *Proceedings of ACM MobiCom*, 2017.

[19] X. Zheng, Z. Cao, J. Wang, Y. He, and Y. Liu, "Zisense: towards interference resilient duty cycling in wireless sensor networks," in *Proceedings of ACM SenSys*, 2014.

[20] 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 ACM IPSN*, 2013.

[21] X. Zheng, Z. Cao, J. Wang, Y. He, and Y. Liu, "Interference resilient duty cycling for wireless sensor networks under co-existing environments," *IEEE Transactions on Communications*, vol. 65, no. 7, pp. 2971–2984, 2017.

[22] Z. Zhao, W. Dong, G. Chen, G. Min, T. Gu, and J. Bu, "Embracing corruption burstiness: Fast error recovery for zigbee under wi-fi interference," *IEEE Transactions on Mobile Computing*, 2016.

[23] J. Wang, Z. Cao, X. Mao, X.-Y. Li, and Y. Liu, "Towards energy efficient duty-cycled networks: Analysis, implications and improvement," *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 270–280, 2016.

[24] W. Joseph, D. Pareit, G. Vermeeren, D. Naudts, I. Moerman, and L. Martens, "Duty cycles of wireless applications and activities for wifi exposure assessment," in *Joint Meeting of the Bioelectromagnetics Society and the European BioElectromagnetics Association*, 2013.

[25] M. R. Oularbi, A. Aissa-El-Bey, and S. Houcke, "Physical layer ieee 802.11 channel occupancy rate estimation," in *Proceedings of IEEE ISVC*, 2010.

[26] A. Botta, A. Dainotti, and A. Pescapé, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012.

[27] Z. Li, Y. Xie, M. Li, and K. Jamieson, "Recitation: Rehearsing wireless packet reception in software," in *Proceedings of ACM MobiCom*, 2015.