XFi: Cross-technology IoT Data Collection via Commodity WiFi

Ruofeng Liu*, Zhimeng Yin*, Wenchao Jiang†, Tian He*
*University of Minnesota
†Singapore University of Technology and Design
liux4189@umn.edu

***Abstract*—Wireless technologies are increasingly diversified to serve various Internet-of-things applications. Yet, our mobile devices (e.g., smartphones) are manufactured with limited types of wireless radio, making it challenging to access the data in the heterogeneous IoT devices. To address this fundamental problem, this work proposes XFi, which enables mobile devices to use *commodity* WiFi radio to directly and simultaneously collect data from diverse heterogeneous IoT devices. Our critical insight is that when an IoT frame collides with an ongoing WiFi transmission, its IoT data is captured by WiFi receiver and retained even after the demodulation procedures in WiFi hardware. Motivated by this observation, XFi proposes a general approach to obtain IoT data by analyzing the decoded WiFi payload. The method is fully compatible with existing commodity WiFi hardware and generally applicable to various IoT protocols. We implement XFi on commodity devices (e.g., RTL8812au, CC2650, and SX1280). Our comprehensive evaluation demonstrates that XFi can collect data from 8 IoT devices in parallel with over 97% accuracy, offering reliable cross-technology data collection.**

***Index Terms*—WiFi, ZigBee, LoRa, Internet-of-Things**

I. Introduction

Internet of things (IoT) are rapidly expanding to every aspect of our daily life with the unprecedented proliferation of heterogeneous wireless devices (e.g., ZigBee sensors, LoRa smart meters, Bluetooth smart wearable, and wireless baby monitors). Naturally, people desire to use their mobile devices (e.g., smartphones) to interact with the diverse IoT devices, manage IoT data, and conduct data fusion operations. However, the increasing diversity of wireless technologies adopted by IoT devices brings about a unique challenge: due to the constraints in the size and complexity, mobile devices are manufactured with limited types of wireless radios such as WiFi and Bluetooth, which inhibits them from directly collect data from heterogeneous IoT devices (e.g., ZigBee and LoRa).

To alleviate this issue, we study the possibility to extend the pervasively available radio in mobile devices, i.e., WiFi, to support heterogeneous IoT techniques in the overlapping ISM spectrum. By doing this, we envision that with a moderate software upgrade, mobile devices can collect data from various types of heterogeneous IoT devices directly, which paves the way for a lot of novel IoT applications.

However, it is technically challenging to enable data exchange between WiFi and heterogeneous IoT radio due to their distinct physical layers. Recent advances in cross-technology communication [9], [30], [31] propose signal emulation, i.e., manipulating high-speed WiFi radio to emulate low-speed ZigBee signal, which allows WiFi to deliver messages to ZigBee. Nevertheless, this series of techniques are not applicable to the communication from low-speed IoT radios back to WiFi because IoT radios generally have very limited capabilities - it is impossible for a ZigBee radio with a 2 MHz bandwidth to produce a 20 MHz WiFi signal.

To address this problem, there are a few previous attempts [18], [39]. Yet, they are fundamentally limited in *compatibility* and *generality*. Specifically, they need to modify the WiFi demodulation procedure for customized decoding of raw IoT signal, making them incompatible with billions of existing commodity WiFi radio in the smartphone. In addition, they are inherently restricted to only support ZigBee without serving other IoT techniques that are being increasingly diversified.

This paper presents XFi - the first design that achieves data collection from *diverse* heterogeneous IoT devices using *commodity* WiFi radio with only a software upgrade. The basic idea is signal hitchhiking - when a smartphone is receiving a WiFi packet from an AP, IoT devices transmit simultaneously, leading to intentional collisions with the WiFi packet in the air. In this way, low-speed IoT data hitchhikes on the high-speed WiFi packet and enters the WiFi radio of the smartphone.

The most critical question XFi needs to address in this paper is how to obtain the IoT data hitchhiking on the received WiFi packet. With our goal of serving diverse IoT techniques with distinct physical layers in minds, XFi proposes a general approach: *reconstruct and decode*, i.e., 1) reconstruct the I/Q waveform of hitchhiking IoT data and then 2) decode the reconstructed IoT waveform. Although the idea sounds straightforward, it is extremely challenging to accomplish in a commodity WiFi device with only a software upgrade. In particular, commodity WiFi hardware does not expose raw I/Q data to the software - It only returns the decoded WiFi payload after WiFi demodulation. To reliably retrieve IoT data from the decoded payload, XFi has to tackle several critical challenges imposed by WiFi demodulation procedures.

- Waveform Reconstruction: Commodity WiFi receiver applies forward error correction (FEC) algorithm to correct errors incurred by wireless interference. Consequently, the hitchhiking IoT signal could be considered by FEC as interference and eliminated amid error correction, rendering it challenging to reconstruct IoT waveform from the decoded WiFi payload. Counter-intuitively, we

978-1-7281-6992-7/20/$31.00 ©2020 IEEE

observe that IoT waveform is retained even after error correction, so that XFi manages to reconstruct a large portion of IoT waveform from the decoded payload.

- Robust Decoding: Commodity WiFi receiver discards informationless segments of WiFi signal (e.g., guard intervals between WiFi symbols). Thus, IoT signal hitchhiking on these discarded segments cannot be recreated in the reconstructed waveform. To cope with the challenge and reliably obtain IoT data, we enhance IoT decoders, which provides them robustness against discarded waveform.

Importantly, our proposed methods are generally applicable to various types of IoT techniques, which is essential for our motivation for supporting diversified IoT devices. We demonstrate the applicability to two representative IoT techniques using highly distinct physical layers: ZigBee and LoRa while they can be extended to other techniques.

In summary, our intellectual contributions are as follows:

- XFi is the *first* work that enables data collection from *diverse* IoT techniques with a *commodity* WiFi radio, without compromising WiFi demodulation.
- Our design simultaneously features generality to various IoT techniques and compatibility with commodity WiFi hardware. To achieve this, XFi obtains IoT data only using decoded payload while tackling critical challenges, e.g., waveform construction and robust decoding.
- We implement XFi on COTS WiFi devices (e.g., RTL8812au) and evaluate XFi across various scenarios. The results demonstrate that XFi can concurrently receive 2 streams of ZigBee or 8 streams of LoRa with 97% accuracy while reaching a throughput of 1.8 Mbps.

II. MOTIVATION

This section presents the motivation of cross-technology IoT data collection using commodity WiFi radio in a smartphone.

Deficiency of IoT Support in Smartphone: Internet of Things (IoT) has been widely deployed, where a large number of heterogeneous IoT devices are generating more than 500 zettabytes IoT data per year [13]. A natural way for people to interact with these ubiquitous IoT devices is through their mobile devices (e.g., smartphones), which brings several unique benefits. For example, a smartphone can provide a friendly user interface for visualizing and managing IoT data, while the powerful computing capability of the smartphone also offers an ideal platform for analyzing heterogeneous IoT data through sensor fusion and machine learning operations.

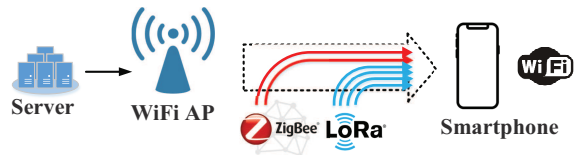
The challenge, however, is that wireless techniques adopted in IoT devices are increasingly diversified while our mobile device is constrained to provide limited options of wireless techniques. For instance, environmental sensors are commonly equipped with ZigBee, while smart meters could adopt LoRa. There are also a large number of IoT devices (e.g., health wearable and baby monitors [34]) that use proprietary wireless protocols. In contrast, restricted by the size and complexity, a smartphone is only equipped with radios such as WiFi, while cramming additional IoT interfaces (e.g., ZigBee) into smartphones [14] has been proved commercially inviable.

Conventionally, a smartphone has to indirectly access the data in heterogeneous IoT devices via extra IoT gateways [3]. However, the dedicated gateway incurs additional hardware cost, deployment complexity, and administration burden, which prevents it from being ubiquitously deployed. In addition, an IoT gateway with a specific wireless radio cannot serve other emerging IoT techniques. As a result, there is an emergent need for a low-cost and general method for the heterogeneous IoT data collection on mobile devices.

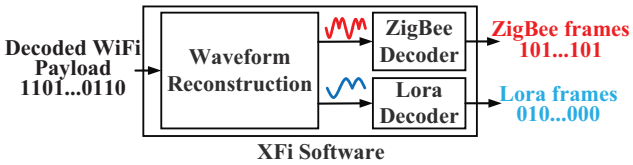
Cross-technology Data Collection via Commodity WiFi: We propose to directly collect data from heterogeneous IoT devices with commodity WiFi radio available in every smartphone. With a moderate software upgrade, smartphones can access diversified IoT devices without extra gateways.

Although direct communications from heterogeneous IoT radio to WiFi have been studied in recent works [18], [39], they commonly work on physical layers requiring a redesign of WiFi decoder. Therefore, they are incompatible with commodity WiFi hardware in mobile devices. Furthermore, their designs are tightly coupled with unique features of ZigBee signal, which cannot be generalized to other types of IoT techniques that are increasingly diversified.

In XFi, we simultaneously address the challenges in both compatibility and generality, enabling mobile devices to obtain IoT data of various heterogeneous techniques using only the decoded WiFi payload that is accessible from WiFi software.



(a) XFi Scenario



(b) XFi Software Architecture

Fig. 1: (a) XFi Enables a WiFi Smartphone to Collect Data from Heterogeneous IoT Devices. (b) XFi Analyzes the Decoded WiFi Payload to Obtain ZigBee or LoRa Data.

III. XFI IN A NUTSHELL

Overview. XFi enables a commodity WiFi radio in mobile devices to collect data from diversified narrowband IoT devices (e.g., ZigBee and LoRa[1]) with only a software upgrade. The high-level idea is signal hitchhiking demonstrated in Fig.1(a): a mobile device downloads a standard WiFi data packet from the associated WiFi AP. When the WiFi packet is being delivered by AP, IoT devices send data in the overlapped spectrum such that low-speed IoT data can hitchhike on the

[1]Note that while LoRa initially operates on the sub-GHz band, 2.4GHz LoRa transceivers (e.g., SX1280/SX1281) have been deployed since 2017.

high-speed WiFi packet in the air and enter the WiFi radio of the mobile device. In the rest of paper, we will refer to ZigBee or LoRa radios as "IoT devices" and ZigBee or LoRa data frames as "IoT frames".

Applicable scenarios. XFi is applicable in a wide range of scenarios since WiFi APs have been ubiquitously deployed for Internet service. For example, over 79% US households get WiFi at home [6] while there are 432 million public APs worldwide [21]. Additionally, XFi is also possible to be applied in mobile scenarios without WiFi APs. It is an increasingly common situation that people carry more than one mobile device. For instance, 60 million US people are using smartwatches in addition to smartphones [15], which can serve as WiFi transmitters with only a software upgrade.

System architecture. The critical component of the system is XFi software installed in mobile devices. As depicted in Fig.1(b), it is able to obtain the hitchhiking IoT data without accessing raw I/Q signal. Specifically, XFi software only uses the decoded WiFi payload returned by commodity WiFi hardware as the input. To retrieve IoT data, it first reconstructs IoT waveform by analyzing the decoded WiFi payload (Section §V) and then reliably decodes the reconstructed IoT waveform with our enhanced IoT decoders (Section §VI).

XFi has three unique features:

- **Compatibility**: XFi retrieves heterogeneous IoT data only via the decoded payload available in WiFi software. Therefore, it is fully compatible with commodity WiFi devices and does not require any hardware modification.
- **Generality**: XFi manages to reconstruct raw RF waveform of hitchhiking IoT data regardless of the wireless technique it uses. This capability makes XFi generally applicable to various IoT protocols that coexist with WiFi on ISM band. We demonstrate the generality with two popular techniques (i.e., ZigBee and LoRa) with highly different physical layers.
- **Efficiency**: XFi allows the wide-band WiFi receiver to concurrently receive and decode multiple heterogeneous narrowband IoT transmissions, significantly improving the network efficiency of densely-deployed IoT devices. Taking Fig.1(a) for example, XFi is able to simultaneously decode the transmissions from two ZigBee devices and four LoRa devices allocated at different channels.

IV. Background and Challenges

This section introduces the background of commodity WiFi receiver and then analyzes the challenges of obtaining IoT data under the constraints of WiFi demodulation procedures.

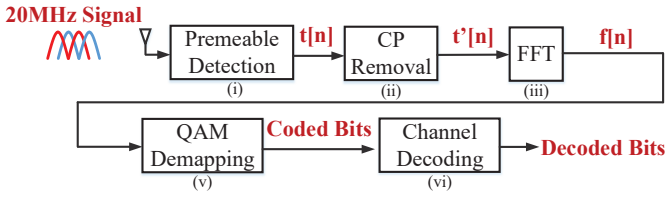


Fig. 2: Demodulation Procedures in WiFi Receiver.

A. Background

In Fig. 2, we illustrate how the WiFi receiver works. (i) Preamble detection searches the predefined training signal, i.e., WiFi preamble in the received 20 MHz signal. If the preamble is detected, WiFi receiver conducts the following OFDM demodulation procedures. Otherwise, the signal is rejected. (ii) For a signal $t[n]$ that passes preamble detection, the cyclic prefix (CP), i.e., the 0.8us at the beginning of each 4us WiFi symbol is removed because it is an "informationless" guarding interval between WiFi symbols for eliminating the inter-symbol interference. (iii) The surviving time-domain waveform $t'[n]$ is transformed into the frequency domain by Fast Fourier Transform (FFT). Each WiFi symbol is transformed into 64 frequency components that correspond to 64 WiFi subcarriers over 20 MHz WiFi spectrum. (v) The QAM demapping discretizes the continuous-value frequency components into bits. Then bits on all the subcarriers are serialized to a sequence of coded bits (or a codeword). (vi) The channel decoder corrects errors in coded bits using forward error correction (FEC) algorithm and produces decoded bits that are available in WiFi software. Note The corrupted decoded bits with CRC check failure by default are dropped by low-level WiFi software for efficiency. However, we can intentionally access these data by editing the filter in the driver.

B. Challenges

The top of Fig.3 shows signal hitchhiking where the input signal to WiFi receiver is 20 MHz WiFi signal with one or more narrowband IoT frames that hitchhike on it. Since XFi is compatible with commodity WiFi hardware, the mixed signal is demodulated through the WiFi decoding pipeline, i.e., (i)-(vi) in Fig. 2, which produces the decoded bits. Our objective is to use these decoded bits to reconstruct IoT waveform and then decode the waveform to obtain IoT data. To achieve this, XFi has to address several challenges in a top-down order.

• **Waveform Reconstruction.** Channel decoder adopts forward error correction (FEC) to protect WiFi transmissions against interference. Presumably, FEC considers hitchhiking IoT signal as interference and attempts to eliminate it in the decoded output. Specifically, in the step (vi), the channel decoder performs error correction to recover collision-free WiFi data, which may prevent the reconstruction of IoT waveform using the decoded bits. In Section §V, our in-depth look into the channel decoding algorithm reveals a counter-intuitive phenomenon that IoT waveform can survive channel decoding because it creates an excessive number of errors.

• **Robust Decoding.** While a large portion of IoT waveform can be reconstructed from the decoded payload, there are still a lot of waveform segments erased amid WiFi demodulation. For example, CP removal in the step (ii) erases the IoT waveform hitchhiking on the cyclic prefix, causing problems to IoT decoders. To reliably decode the partially reconstructed waveform, Section §VI enhances decoder's robustness against signal erasure by leveraging a unique signal erasure pattern.

• **Other Challenges.** XFi has to address other practical challenges. For example, WiFi receiver only decodes after

receiving WiFi preamble. For signal hitchhiking, IoT devices need to know when to transmit in order to be simultaneous with WiFi transmitter, which is discussed in Section §VII.

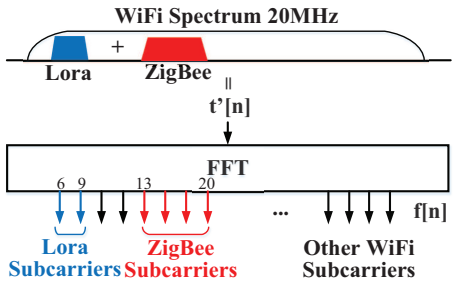


Fig. 3: FFT of WiFi Signal with Hitchhiking IoT Data.

V. Waveform Reconstruction

This section presents the design to reconstruct the raw waveform of hitchhiking IoT data from decoded WiFi payload.

A. Channel Decoding and Error Correction

As an OFDM receiver, a commodity WiFi receiver converts the time-domain signal into the frequency domain via FFT, so that it can demodulate the received data at each subcarrier. Fig.3 demonstrates that each narrowband IoT signal interferes with the WiFi data in the specific subcarriers. For example, ZigBee signal disturbs subcarrier 13 to 20, while LoRa signal disturbs subcarrier 6 to 9. So ideally, XFi can examine the changes of values on the corresponding subcarriers to reconstruct each hitchhiking IoT signal.

However, channel decoder performs error correction (FEC) on the received data to get rid of narrowband interference, which could potentially recover the original value on the subcarriers and eliminate hitchhiking IoT signal. In Fig. 4, we use a toy example to illustrate the error correction of Low Density Parity Check (LDPC) decoder in WiFi. Specifically, the received data on the subcarriers are first mapped to coded bits consisting of "data bits" (d_0, d_1, d_2) and "parity bits" (p_0, p_1, p_3). The hitchhiking ZigBee signal disturbs the data bit d_1 such that d_1 is changed to an incorrect value: 0.

While we desire to infer the IoT data from such errors, FEC comes in to detect and correct them. As the top of Fig. 4 shows, data bits d_1, d_2, and parity bits p_1 are connected to a common parity check constraint for error detection. The binary parity check fails since the summation in $gf(2)$, i.e., $d_1 \oplus d_2 \oplus p_1 \neq 0$. Similarly, the other two parity checks that d_1 is associated with also fail. Therefore, the parity checks come to a consensus that d_1 is incorrect, so LDPC decoder might flip d_1 for the parity check constraints to be satisfied. If this happens, error correction might unfortunately eliminate the influence of ZigBee waveform in the coded bits.

It is noteworthy that the real coded bits in WiFi are much longer than the example (typically 1944) with the 324 parity checks. Thus, the decoder needs to repeat the procedure above multiple times for the coded bits to converge. Finally, WiFi versions targeted by the paper include 802.11n/ax/ac, which are supported in majority of commodity mobile devices.

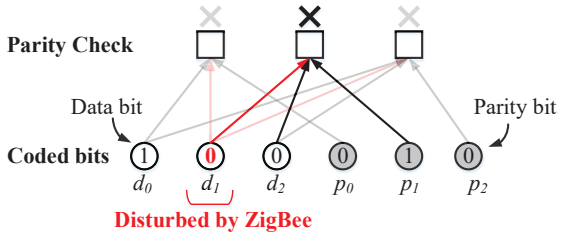


Fig. 4: Error Correction in LDPC Decoder.

B. Feasibility of Waveform Reconstruction

Counter-intuitively, we observe that LDPC decoder almost keeps the corrupted coded bits intact. In specific, when coded bits are severely disturbed by heterogeneous IoT signal, channel decoder only flips an extremely limited number of "coded bits" ($\leq 2\%$). Our key insight for this phenomenon is that the channel decoding algorithm has an *Error Correction Capability*. When received coded bits have errors that exceed the algorithm's capacity, parity checks fail to reach a consensus and the decoder becomes ineffective in correcting errors.

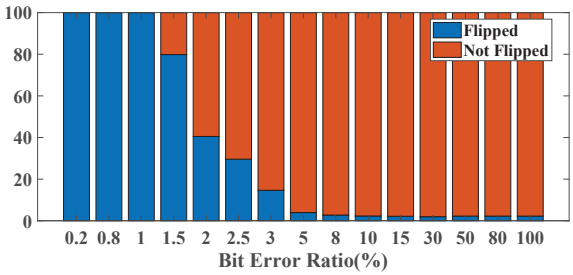


Fig. 5: Error Correction (Flip) Ratio vs. Bit Error Ratio.

In the section, we demonstrate this observation with an empirical experiment while the theoretical proof is given in the Appendix. Specifically, we first generate a valid WiFi codeword and then purposely inject varying ratios of bit errors into the codeword such that it gradually moves away from the correct value. We use LDPC decoder to decode each modified WiFi codeword and calculate the percentage of errors that are corrected (flipped) to examine the effectiveness of LDPC decoder under different bit error ratios.

The result in Fig. 5 shows that LDPC decoder is able to perform with a small bit error ratio (e.g., 1.5%), while it becomes ineffective and only flips 1.8% bits when the bit error ratio exceeds 5%. The interference created by hitchhiking IoT signal (e.g., ZigBee and LoRa) is sufficient to make LDPC decoder ineffective. For example, ZigBee has a 2 MHz bandwidth, which is equivalent to 15% of bit errors, while LoRa with a 0.8 MHz bandwidth incurs around 7% of bit errors. Consequently, channel decoder preserves IoT signal hitchhiking on each subcarrier, so that XFi can reconstruct the IoT waveform by elaborately analyzing decoded bits. Since these decoded bits are available in software, our design can be compatible with commodity WiFi hardware.

C. Waveform Reconstruction Algorithm

In this section, we present the details of reconstructing IoT waveform with decoded bits. As Fig.6 demonstrates, waveform

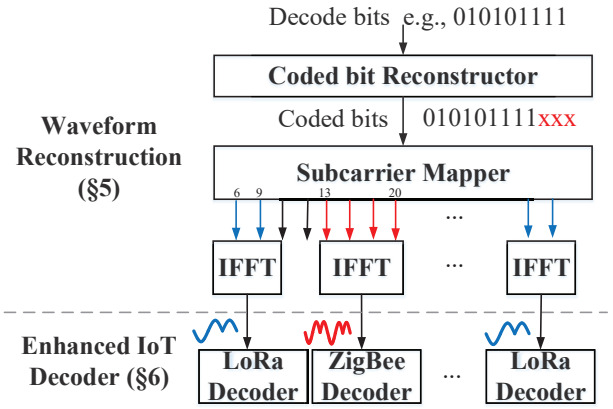


Fig. 6: XFi Architecture.

reconstruction is a general module that recreates IoT signal of various techniques in the separated channels by analyzing the decoded bits returned to WiFi software. The reconstructed IoT waveforms are then demodulated using the enhanced IoT decoders to obtain IoT data, which we will discuss in §VI.

Specifically, XFi first approximates coded bits with decoded bits. As discussed in §V-B, channel decoder becomes ineffective due to the excessive interference from IoT signal, so decoded bits provide a close approximation of coded bits. Second, coded bits are mapped to frequency domain, i.e, subcarriers. Finally, each individual IoT signal resides in the separated narrowband, so XFi aggregates the changes of values on the corresponding subcarriers and then performs inverse fast fourier transform (IFFT) to reconstruct IoT waveform. For example, when a ZigBee frame is at the frequency range from -4 MHz to -6 MHz w.r.t the center frequency of WiFi, XFi aggregates the value changes on 13^{th} to 20^{th} subcarriers and performs IFFT to compute the time-domain ZigBee waveform.

Note that decoded bits do not contain the parity bits which are dropped by WiFi receiver as we will further discuss in §VI-A1. In Fig.6, we mark unknown parity bits as "x" when approximating coded bits and the corresponding IoT waveform cannot be reconstructed. We tackle the challenge in §VI.

VI. Robust Decoding

Waveform construction effectively recovers a large portion of IoT waveform. Yet, there are still a lot of waveform segments that are erased amid WiFi demodulation, imposing challenges to obtain the hitchhiking IoT data. This section first analyzes two major sources of signal erasure (i.e., Parity and CP). Then we introduce our enhanced IoT decoders that provide robustness against signal erasure in the software.

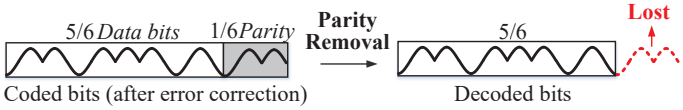


Fig. 7: Parity Removal Erasures IoT waveform.

A. Signal Erasure

1) Parity Removal: As we mention in §V-A, LDPC decoder takes the data bits and parity bits as input, and then tries to correct errors in data bits with parity bits. After that, it discards the parity bits and only passes data bits to the software.

Although we observe that the error correction has negligible impacts, the removal of parity bits inevitably leads to signal erasure because parity bits also contain IoT signal. As Fig. 7 depicts, when the coding rate of LDPC is $\frac{5}{6}$, $\frac{1}{6}$ of coded bits are parity bits. Consequently, $\frac{1}{6}$ of IoT signal is erased and cannot be accessed by XFi via decoded bits.

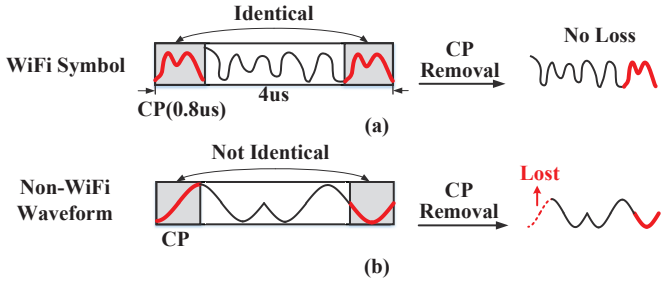


Fig. 8: (a) CP Removal of WiFi Symbol. (b) CP Removal of IoT Waveform Erasures IoT waveform.

2) CP Removal: Besides parity removal, CP removal also incurs signal erasure. As depicted in Fig. 8(a), for each WiFi symbol with a duration of $4us$, WiFi receiver drops the guarding signal in the first $0.8us$, which is called cyclic prefix (CP). In a standard WiFi symbol, the transmitted waveform at CP is identical to the last $0.8us$ of a symbol, suggesting that the WiFi receiver will not lose any information for removing CP. In contrast, due to heterogeneity between wireless protocols, non-WiFi signal (e.g., ZigBee) does not have this CP feature, as we demonstrate in Fig.8(b). As a result, when WiFi receiver removes "CP" in IoT signal every $4us$, a segment of $0.8us$ IoT signal is lost, imposing challenges for obtaining IoT data.

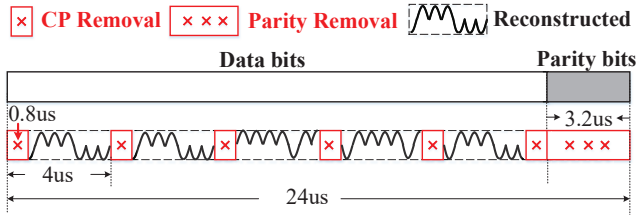


Fig. 9: Overall Pattern of Signal Erasure.

3) Summary of Signal Erasure Pattern: The overall signal erasure pattern is illustrated in Fig.9 where $\frac{1}{3}$ of IoT waveforms are erased by WiFi hardware. The erased waveform partitions are marked by red 'x', which can not be accessed from decoded bits. Specifically, In every $24us$, XFi suffers from the one $3.2us$ signal erasure due to parity removal. In addition, XFi also suffers from $0.8us$ signal erasure due to CP removal every $4us$. These two factors are the dominating waveform loss in XFi, while XFi also suffers from the quantization error when QAM demapping discretizes subcarriers and the bit flippings ($< 1.8\%$) during error correction (§V-B).

Our evaluation (§IX) shows both quantization and bit flippings have limited impacts and do not lead to decoding errors.

To reliably decode the reconstructed waveform with signal erasure, XFi customizes the IoT decoder to incorporate the signal erasure pattern with the symbol-level (§VI-B) and chip-level (§VI-C) redundancy of IoT signal. To make our description concrete, our discussion focuses on ZigBee while the idea generally applies to other techniques (e.g., LoRa). In §IX, we provide evaluation results for both techniques.

B. Combat Signal Erasure: Symbol-level

ZigBee symbols contain redundant information for the protection against interference In the section, we combine the signal erasure pattern observed in §VI-A3 with this redundancy to offer ZigBee decoder robustness against signal erasure. Note that symbol-level redundancy is widely provided by various IoT techniques, e.g., LoRa spreads symbols into chirps. Although our description focuses on ZigBee, this idea is applicable to other IoT protocols.

1) ZigBee Symbol: ZigBee adopts direct sequence spread spectrum (DSSS) which spreads each ZigBee symbol (4 bits) into 32 binary chips following the DSSS table (Table 1). This inherent redundancy introduced by DSSS helps XFi to decode partially reconstructed ZigBee signal, as discussed later. In a standard ZigBee receiver, a set of 32 ZigBee chips are then mapped to a ZigBee symbol by comparing the hamming distances between the demodulated chip sequence and the 16 standard DSSS sequences in Table 1. The symbol with the smallest hamming distance is selected.

Symbol (4 bits)	Chip Sequence (32 bits)
0 0 0 0	11011001110000110101001000101110
0 0 0 1	11101101100111000011010100100010
...	...
1 1 1 1	11001001011000000111011110111000

TABLE I: Symbol-to-chip Mapping in ZigBee (802.15.4)

2) Erasure-Aware Symbol Decoding: As analyzed in §VI-A, the reconstructed IoT waveform suffers from the signal erasure in specific sections, i.e., CP and parity. This knowledge motivates us to design an erasure-aware DSSS decoding algorithm adapted to the unique signal erasure pattern.

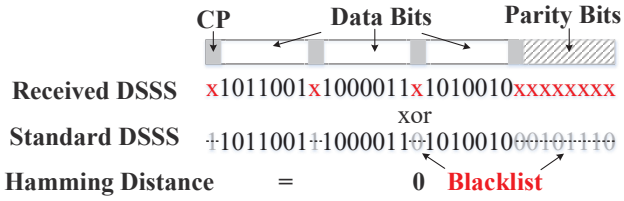


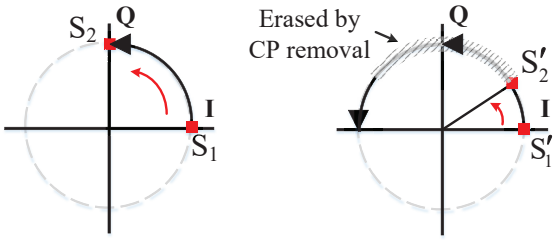
Fig. 10: Decoding ZigBee Symbol with Blacklist.

In specific, XFi can use the signal erasure pattern to identify which ZigBee chips are undetermined and thus cannot be trusted in DSSS decoding. Therefore, when XFi calculates the hamming distance, it blacklists these undetermined chips, and only utilizes the remaining chips for improving symbol decoding reliability. As Fig.10 shows, XFi blacklists the chips which are missing due to CP removal and parity removal, while it only relies on the chips that can be reconstructed from data bits. In this example, the chip sequence has the smallest hamming distance to ZigBee symbol 0, and is thus decoded as symbol 0. By doing this, XFi offers the decoder robustness to missing chips. Note that this procedure is after the detection of ZigBee preamble, which is discussed in §VII-A.

C. Combat Signal Erasure: Chip-level

In addition to symbol-level redundancy, XFi explores the resilience in ZigBee chip to enhance the decoder's robustness.

1) ZigBee Chip: ZigBee chips are modulated by offset quadrature phase-shift keying (OQPSK) where each ZigBee chip is encoded by a $0.5us$ phase shift. Specifically, as Fig.11(a) depicts, chip 1s are modulated by positive phase shifts ($\frac{\pi}{2}$) while chip 0s are represented by negative phase shifts ($-\frac{\pi}{2}$). The standard OQPSK demodulator uniformly samples the signal every 0.5us (e.g., S_1 and S_2) and calculates phase shifts between samples.



(a) Standard ZigBee Samples (b) CP-aware ZigBee Samples

Fig. 11: ZigBee Chip. (a) Uniform Samples of Standard ZigBee Chip. (b) Erasure-aware Samples of Partially Reconstructed ZigBee Chip.

2) Erasure-aware Chip Demodulation: ZigBee chips may partially suffer from signal erasure as the dashed line in Fig.11(b) indicates. To decode these chips, XFi introduces the erasure-aware chip decoding by elaborately benefiting from the feature of ZigBee chip. Specifically, in one chip, the phase is monotonically increasing or decreasing, so the sign of phase shift can be calculated by any two distinct samples. Therefore, as Fig.11(b) shows, instead of sampling uniformly in 2 MHz, XFi carefully chooses the two samples, i.e., S_1' and S_2' in the reconstructed waveform to avoid the signal erasure.

VII. Practical Signal Hitchhiking

Section §V and §VI demonstrate the feasibility of decoding IoT signal, when it collides into an ongoing WiFi frame. This section further discusses two additional issues for realizing XFi in practical scenarios. Specifically, §VII-A introduces how to notify IoT devices, so that they could utilize the signal hitchhiking opportunity. In §VII-B, we discuss the design of WiFi payload that maximizes the signal hitchhiking reliability.

A. Coordination

To achieve signal hitchhiking, IoT devices must send data to a WiFi device when the WiFi device is receiving an ongoing frame from WiFi AP. Interestingly, such coordination can be

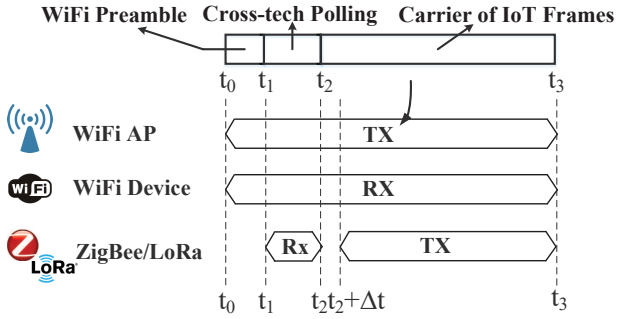


Fig. 12: The downlink WiFi frame from AP consists of a cross-technology polling notification to IoT devices ($t_1 - t_2$) and a carrier of IoT signal ($t_2 - t_3$).

achieved by the WiFi frame itself. More specifically, we elaborately select the WiFi payload such that the corresponding WiFi frame delivered by AP can simultaneously accomplish 3 critical goals: (1) polls IoT devices to send data, (2) carriers IoT data into WiFi receiver, and (3) allocates IoT devices to distinct channels for avoiding collisions.

The WiFi frame structure and the complete signal hitchhiking procedures are demonstrated in Fig.12. First, as a legitimate WiFi frame, the frame starts with WiFi preamble at t_0. Upon the detection of the preamble, the WiFi radio begins receiving and decoding, which provides an opportunity to capture the IoT signal. Second, the segment of WiFi payload from t_1 to t_2 notifies the IoT devices of the opportunity of signal hitchhiking. In specific, XFi adopts WiFi→IoT CTC techniques recently proposed in [30] to embed a legitimate ZigBee or LoRa packet in the WiFi frame that contains a list of IoT devices to be queried and their allocated channels. After receiving the notification, IoT devices tune the transmitter to the allocated channel, temporarily disable collision-avoid mechanism (e.g., CSMA), and then start transmissions at $t_2 + \Delta t$. Narrowband IoT frames collide with the rest part of the WiFi frame, which carries them into the WiFi device.

Note that the WiFi frame and IoT frames do not need to be strictly synchronized, as Fig.12 depicts. XFi reconstructs the complete narrowband signal from t_2 to t_3, which is then match-filtered with the predefined ZigBee or LoRa preamble to detect the beginning of IoT frames (i.e., $t_2 + \Delta t$).

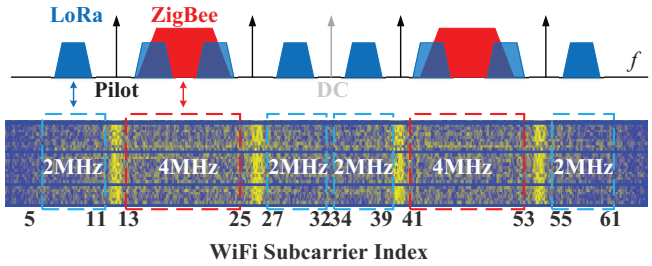


Fig. 13: Power Spectrum and Channel Allocation.

B. Power Control

An issue arises immediately: High energy of WiFi can saturate the WiFi subcarriers, leading to the corruption of IoT frames. To address the problem, XFi introduces a power control mechanism - by elaborately selecting the WiFi payload, we can minimize the WiFi energy on specific subcarriers. In particular, XFi carefully chooses the payload that produces WiFi signal from t_2 to t_3 in Fig.12, so that the subcarriers capturing IoT signal take minimal values. By doing this, XFi effectively minimizes the WiFi energy at the frequency ranges where IoT frames hitchhike. Fig.13 depicts the visual power spectrum density, where blue indicates low energy and yellow refers to high energy. Obviously, several narrowbands with visibly lower energy than other subcarriers (e.g., pilots) are produced, thus improving the IoT signal's signal-to-noise ratio.

With the power control, XFi utilizes WiFi spectrum with extremely high efficiency. As the top of Fig.13 shows, we create two continuous 4 MHz channels and four 2 MHz channels that can pack 2 ZigBee frames or 8 LoRa frames. Besides, XFi carefully allocates guarding bands between channels to mitigate the interferences between channels. The evaluation demonstrates that the power control mechanism effectively carries IoT data over WiFi transmission.

VIII. Limitation and Discussion

Implementation of XFi on smartphones. XFi works in WiFi software, so it is compatible with commodity WiFi hardware in smartphones, requiring only a moderate driver upgrade. As a proof of concept, we prototype and evaluate XFi with commodity WiFi NIC in a Linux PC because it is convenient to patch Linux open-source WiFi drivers and obtain corrupted WiFi frames. WiFi drivers of smartphone are close-source and need be patched with special patch tools (e.g., NexMon [33]), which is left to future works.

Limitation of communication Range: XFi cannot support long range communications with IoT devices, which is because XFi can reconstruct IoT signals only when the signal is strong enough to corrupt WiFi packets. For long ranges, IoT signals cannot significantly interfere with WiFi packets due to the limited transmission power of low-power IoT transmitters and uncertainty of the channel. As a result, IoT signals will be eliminated by error correction of LDPC decoder.

Generalization of XFi. The proposed approach in XFi generally applies to other IoT techniques. To extend XFi to another technique, one can directly adopt signal hitchhiking and waveform reconstruction to capture and reconstruct its waveform while replacing the decoder (Section §VI) to one specific to target technique. Additionally, it is possible to extend XFi to scenarios in other frequency bands, e.g., communication among WiFi 802.11ah and sub-GHz IoT devices.

XFi overhead. XFi introduces little overhead to WiFi and IoT network. Due to WiFi's collision avoidance mechanism, i.e., CSMA, other WiFi traffic will back off when IoT frames are in transmission. Besides, XFi allows parallel data collection from multiple IoT devices to make efficient use of WiFi's wide spectrum. Finally, our coordination design in Section §VII is compatible with the receiver-initiated ZigBee and LoRa protocol (e.g., [35]), so IoT devices can perform duty cycle to minimize the overhead of idle listening.

IX. PERFORMANCE EVALUATION

This section evaluates XFi on commodity IoT and WiFi devices across various scenarios for demonstrating the reliability, efficiency, and generality of XFi.

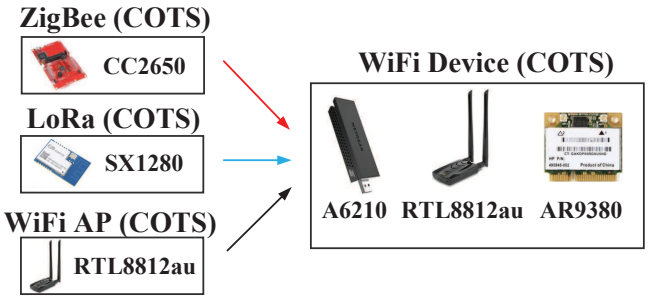


Fig. 14: Experimental Setting for XFi.

A. Evaluation Setup

We prototype XFi on the commercial off-the-shelf WiFi and IoT devices. The evaluation settings are depicted in Fig.14. To demonstrate XFi's compatibility, we evaluated it on Ubuntu 16.04 PC with three representative WiFi chipsets from different brands: Mediatek A6210, Realtek RTL8812au, and Atheros AR9380. WiFi devices operate on channel 3 (2422 MHz). ZigBee performance is evaluated with CC2650 ZigBee SoC [38]. We develop ZigBee program using TI-RTOS SDK. Finally, LoRa is evaluated with SX1280 SoC [36] running default PINGPONG example.

The evaluation starts with the overall performance of XFi. Then the detailed measurements of physical layers (i.e., ZigBee chip error rate and symbol error rate) are provided, followed by the link-layer experiments (i.e., frame reception ratio). To demonstrate the generality of XFi, we also evaluate it on LoRa. Each experiment is repeated 10 times, while 10000 frames are received and the statistical results are obtained. Without further explanation, the distance between WiFi transmitter and receiver is 5 meters and the transmitter adopts 64 QAM modulation, $5/6$ coding rate, and 15 dBm transmission power. The distance between IoT devices and WiFi receiver varies from 2 meters to 15 meters, while the transmission power of IoT devices is 0 dBm.

TABLE II: Summary of XFi Performance.

	Commodity	ZigBee	LoRa
XFi	Yes	285.7 Kbps	1.8 Mbps
Lego-Fi [18]	No	213 Kbps	Not Support
SymBee [39]	No	31.25 Kbps	Not Support
ZiFi [17]	Yes	215.9 bps	Not Support

B. Overall Performance

As we demonstrate in Table II, XFi is the first IoT→WiFi communication design that simultaneously features 1) compatibility with commodity WiFi hardware, 2) generality to various IoT technologies, and 3) high throughput. Specifically, XFi obtains IoT data entirely from decoded WiFi payload, whereas the state-of-the-art designs (e.g., Lego-Fi [18]) require raw signal, which is incompatible with commodity devices. Additionally, in contrast to previous works that are specific to ZigBee, XFi is first to be generally applicable to several distinct wireless techniques (e.g., LoRa), which is extremely beneficial since IoT techniques are increasingly diversified.

With the capability of retrieving the legitimate payload data of multiple IoT frames in parallel, XFi achieves a maximum throughput of 285.7 Kbps for ZigBee and 1.8 Mbps for LoRa. The throughput of XFi significantly outperforms SymBee [39] and ZigFi [17] which rely on coarse-grained information of signal (e.g., correlation and CSI). Note that the latest WiFi standard (i.e., IEEE 802.11ax) provides 160 MHz bandwidth, which allows XFi to supports 16 parallel ZigBee devices and potentially improves the overall throughput by 8 times.

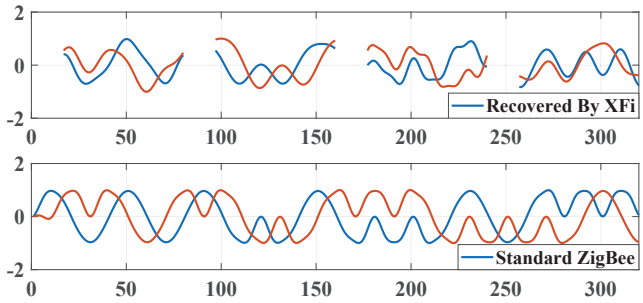
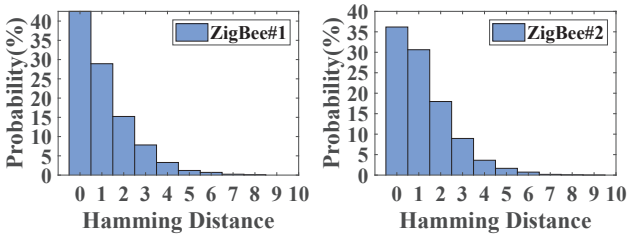


Fig. 15: Reconstructed ZigBee vs. Standard ZigBee.

C. PHY Layer Performance

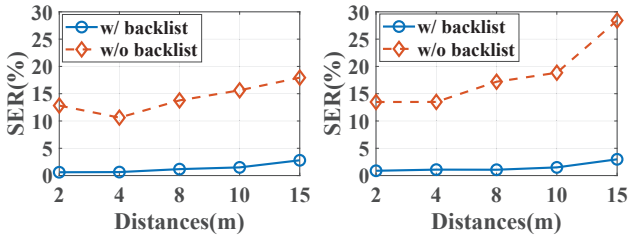
XFi introduces waveform reconstruction (Section §V) to recover raw IoT waveforms from decoded WiFi payload. To evaluate the accuracy, we compare the reconstructed ZigBee signal with the standard one. As Fig.15 illustrates, the narrowband signal recovered by XFi approximates the standard one well for the most of the time except those signal erased by the WiFi receiver amid CP removal and parity removal.



(a) CER: ZigBee Channel 1 (b) CER: ZigBee Channel 2

Fig. 16: Chip Error Distribution in ZigBee Symbols.

We quantify the accuracy of the reconstructed waveform by examining the demodulated chips. In this experiment, two ZigBee devices transmit on parallel channels in Fig.13, while the distance between ZigBee transmitters and WiFi receiver is 2 meters. The distributions of chip errors are depicted in the Fig.16. Over 90% of symbols have less than 3 chips errors, which can be tolerated by DSSS. In addition, these two streams of Zigbee signal demonstrate similar chip error distribution, showing that XFi can obtain multiple ZigBee transmissions from WiFi payload and independently decode them in parallel.

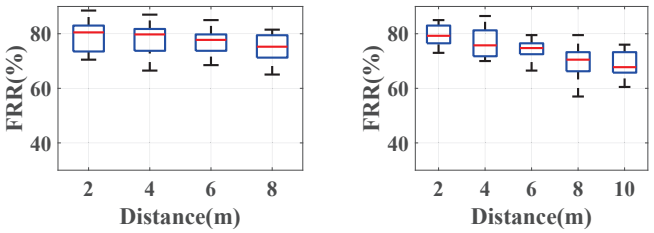


(a) SER: ZigBee Channel 1 (b) SER: ZigBee Channel 2

Fig. 17: Symbol Error Ratio vs. Distances.

D. Symbol Error Rate

This section evaluates the symbol error rate (SER) of ZigBee. To demonstrate the performance gain from our enhancements to the IoT decoder, we compare the symbol error ratios to the ones without our blacklist design (Section (§VI-B). Fig.17 demonstrates that blacklisting undetermined chips can significantly improve the decoding accuracy. For example, in the first channel, it reduces the SER from 12% to 0.61% at 2 meters while it keeps the SER under 5% at 15 meters, which dramatically enhances the robustness of the decoder.



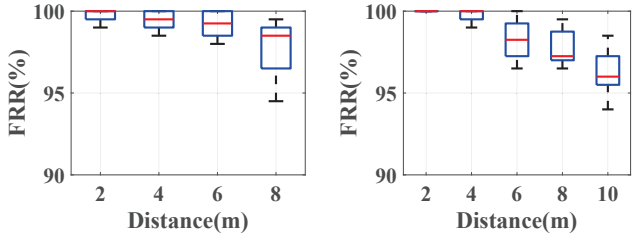
(a) FRR: Lab Room (b) FRR: Corridor

Fig. 18: Frame Reception Ratio vs. Distances.

E. Frame Reception Rate

We evaluate ZigBee frame reception ratios (FRR) in two sites: laboratory room and corridor. Fig.18 plots the average FRR of ZigBee frames with 4 symbols in varying distances. In the noisy lab environment, FRR drops slowly to 75% when increasing distances from 2 to 8 meters, while in the corridor XFi achieves more than 70% FRR at a distance of 10 meters.

To further improve the reliability, we transmit ZigBee frames with 4/7 Reed-Solomon (RS) Code. As demonstrated in Fig.19, XFi achieves $\geq$ 97% FRR in varying sites and distances after the RS code is adopted, showing that XFi can reliably collect data from heterogeneous IoT devices.



(a) FRR w/ Coding (Lab) (b) FRR w/ Coding (Corridor)

Fig. 19: Frame Reception Ratio with RS code.

F. XFi:LoRa

XFi also enables WiFi radio to receive 8 parallel LoRa frames. Note that LoRa uses chirp spread spectrum (CSS) in the physical layer, which is dramatically different from the physical layer of ZigBee, demonstrating that XFi is applicable to various types of IoT techniques. In Fig.20, we depict the spectrum of reconstructed LoRa waveform where LoRa chirps can be identified visibly, which shows the generality of waveform reconstruction to diverse physical layers.

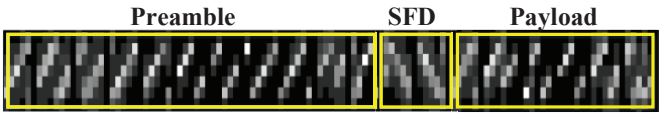
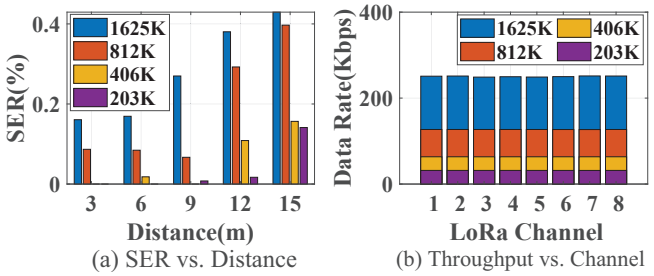


Fig. 20: Spectrum of Reconstructed LoRa Waveform.

Fig.21(a) shows symbol error ratios (SER) of LoRa in various seettings. XFi achieves low error ratio ($< 0.5\%$) within 15 meters. We observe that LoRa signals with a smaller bandwidth are most robust because a longer symbol duration provides more resilience to signal erasure. When the range is further increased, the communications become less reliable. For example, at 25 meters, we receive a lot of WiFi frames containing no errors from which we can reconstruct IoT signal. As we discussed in Section §VIII, IoT radios are limited in signal strength and thus cannot corrupt WiFi frames at a long range. The ranges of XFi is comparable to previous designs such as ZigFi (15 meters), Lego-Fi (20 meters), and SymBee (25 meters). Note that LegoFi and SymBee are implemented on software-defined radios, whereas XFi completely uses commodity WiFi radios. Finally, as Fig.21(b) plots, XFi offers similar reliability and throughput for 8 parallel LoRa channels, significantly improving the spectrum efficiency when LoRa devices are densely deployed.



(a) SER vs. Distance (b) Throughput vs. Channel

Fig. 21: LoRa Performance (Spreading Factor = 5).

X. Related Work

The coexistence and communication among heterogeneous wireless devices have been studied in the literature. The existing works can be categorized as follows.

• **Wireless Coexistence** Traditional researches on wireless coexistence mainly focus on interference avoidance [4], detection [2] and cancellation [5], [27], [29], [32], [45]. XFi is distinct from these works in two aspects. First, XFi purposely leverages interference among WiFi and IoT as an opportunity for cross-technology data collection. Second, these works commonly work in the physical layer, thus requiring access to raw I/Q data and customized receivers. In contrast, XFi uses

the decoded WiFi payload, so it is only a software upgrade that is compatible with commodity WiFi devices.

• **Cross-technology Communication** Cross-technology communication technologies (CTC) enable direct data exchanges among heterogeneous wireless techniques. Early CTC designs manipulate sparse packet-level information (e.g., the packet duration [8], [20], [43], interval [26], energy pattern [10], [16], [23], [41], and energy level [11], [17], [19], [44]) to deliver messages among heterogeneity. Among these works, B2W2 [11] and ZigFi [17] enable low-power IoT radios to send messages to WiFi by intentionally interfering WiFi CSI. However, due to the sparsity of CSI information, these designs are intrinsically restricted in the data rate.

XFi belongs to recent advances of physical-layer CTC [9], [12], [30], [31], [40], [42], which significantly improve the data rate by directly delivering messages via payload. The pioneering work (WEBee [30]) introduces signal emulation that enables WiFi radio to send legitimate ZigBee frames. Despite its success, signal emulation only applies to high-speed radios (e.g., WiFi), so it cannot enable a WiFi radio to collect data from ZigBee. To tackle the problem, SymBee [39] and LEGO-Fi [18] propose to decode ZigBee signal in the WiFi device. However, they modify WiFi demodulation procedures, thus cannot be deployed on commodity WiFi hardware. Besides, their designs are tightly coupled with the unique features of ZigBee signal and thus cannot be generalized to other wireless techniques (e.g., LoRa). In contrast, XFi is the first "IoT→WiFi" CTC that entirely uses commodity WiFi radios and our "reconstruct and decode" technique is also the first general method that can be extended to other IoT technologies. CTC among various IoT technologies (e.g., ZigBee, BLE and LoRa) [22], [24], [28], [37] and CTC between WiFi and LTE [7] are also proposed. These works are either based on signal emulation or unique features of these wireless protocols. Therefore, they cannot be generalized to "IoT→WiFi CTC" (i.e., address communication from low-speed radio to high-speed WiFi). They also do not address the unique challenges of a WiFi radio (e.g., channel decoding and signal erasure). Finally, using mobile devices as IoT gateways is discussed in [1], which only considers homogeneous wireless techniques.

XI. Conclusion

This paper proposes XFi, the first work that achieves cross-technology data collection using commodity WiFi hardware. We envision that XFi will inspire ubiquitous interactions between mobile devices and heterogeneous IoT systems.

XII. Acknowledge

This work was supported by the NSF CNS-1525235, NSF CNS-1718456, NSF CNS-1717059 and SRG ISTD 2020159. We sincerely thank the shepherd and reviewers for their valuable comments.

Appendix

This section theoretically proves our observation in the section V-B that LDPC decoder is ineffective when decoding hitchhiking IoT signal that incurs excessive bit errors.

Proof. The proof is based on sum-product algorithm [25], which is commonly used in LDPC decoder for its high efficiency. Sum-product calculates the a posteriori log-likelihood ratio (LLR) L_i for each bit i as defined in Equation 1 where bit i is decided to be one when $L_i \leq 0$ and zero when $L_i \geq 0$.

$$L_i = log\frac{p(x_i=0)}{p(x_i=1)} = r_i + \sum_{j\in A_i} E_{j,i} \tag{1}$$

L_i is the sum of r_i (the initial LLR from the input) and $E_{j,i}$ (the extrinsic LLR from the j^{th} parity check to bit i). Since a bit is flipped when the sum of extrinsic LLRs is large enough so that the initial LLR (r_i) and a posteriori LLR (L_i) take different signs, the probability of flip (P_e) can be expressed as a conditional probability in Equation 2. The decoder is ineffective if P_e is extremely small for any bit.

$$P_e = P(L_i \leq 0 | r_i \geq 0) = P(L_i \geq 1 | r_i \leq 1) \tag{2}$$

As Equation 3 shows, the initial LLR r_i only depends on the received value of bit i denoted as $\hat{x}_i$, where the crossover probability p is prior knowledge of the decoder.

$$r_i = log\frac{P(x_i=0)}{P(x_i=1)} = \begin{cases} log\frac{p}{1-p} & \hat{x}_i = 1 \\ log\frac{1-p}{p} & \hat{x}_i = 0 \end{cases} \tag{3}$$

Since $r_i \geq 0$ is equivalent to $\hat{x}_i = 0$ and vice versa, P_e in Equation 2 can be deduced into Equation 4.

$$\begin{aligned} P_e &= P(r_i + \sum_{j\in A_i} E_{j,i} \leq 0 | \hat{x}_i = 0) \\ &= P(\sum_{j\in A_i} E_{j,i} < log\frac{p}{1-p}) \end{aligned} \tag{4}$$

$log\frac{p}{1-p}$ is constant, so the calculation of P_e is reduced to find the distribution of $\sum_{j\in A_i} E_{j,i}$. As Equation 5 shows, each extrinsic LLR $E_{j,i}$ is computed from all bits associated with j^{th} parity check except for bit i (denoted as bit i'), where $p_{i'} = P(x_{i'} = 1)$ is the probability that the true value of bit i' is one.

$$E_{j,i} = log(\frac{\frac{1}{2} + \frac{1}{2}\prod_{i'\in B_j, i'\neq i}(1-2p_{i'})}{\frac{1}{2} - \frac{1}{2}\prod_{i'\in B_j, i'\neq i}(1-2p_{i'})}) \tag{5}$$

Our key insight is that coded bits in severely corrupted WiFi payload are almost uncorrelated from each other, meaning that p_i's are $i.i.d$ random variables that take value either p or $1-p$ with $\frac{1}{2}$ probability. Therefore, $E_{j,i}$ is simply a discrete random variable shown in Equation 6, where e is a constant calculated as $e = log(\frac{\frac{1}{2}+\frac{1}{2}\prod_{i'\in B_j, i'\neq i}(1-2p)}{\frac{1}{2}-\frac{1}{2}\prod_{i'\in B_j, i'\neq i}(1-2p)})$.

$$f(E_{j,i}) = \begin{cases} \frac{1}{2} & E_{j,i} = e \\ \frac{1}{2} & E_{j,i} = -e \end{cases} \tag{6}$$

Importantly, $e \ll log\frac{p}{1-p}$. Typically, $p = 0.9$ and $log\frac{p}{1-p}$ = 2.2, while e is less than 0.001. Since corrupted coded bits are highly uncorrelated, $E_{j,i}$s are approximately independent of each other and can also be modeled as $i.i.d$. Thus, the sum of extrinsic LLR is very unlikely to be large enough to flip the sign of LLR. When $p = 0.9$, $P_e \approx 0.1\%$. Note that the experiment shows a slightly higher probability because $E_{j,i}$s are not completely independent. Our observation in Section §V-B is proved. □

REFERENCES

[1] M. Abdelaal, M. Dandy, F. Durr, K. Rothermel, and M. Abdelgawad. Gaas: Adaptive cross-platform gateway for iot applications. In *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 217–226, 2019.

[2] F. Adib, S. Kumar, O. Aryan, S. Gollakota, and D. Katabi. Interference alignment by motion. In *Proceedings of ACM MobiCom, 2013*, 2013.

[3] advantech.com. advantech IoT Gateway. http://select.advantech.com/iotgateway/, 2019.

[4] G. Baig, I. Kash, B. Radunovic, T. Karagiannis, and L. Qiu. Interference management for unlicensed users in shared cbrs spectrum. In *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, pages 333–345, 2018.

[5] T. Bansal, B. Chen, P. Sinha, and K. Srinivasan. Symphony: Cooperative packet recovery over the wired backbone in enterprise wlans. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 351–362. ACM, 2013.

[6] P. R. Center. Internet/Broadband Fact Sheet. https://www.pewresearch.org/internet/fact-sheet/internet-broadband/, 2019.

[7] E. Chai, K. Sundaresan, M. A. Khojastepour, and S. Rangarajan. Lte in unlicensed spectrum: Are we there yet? In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 135–148. ACM, 2016.

[8] K. Chebrolu and A. Dhekne. Esense: communication through energy sensing. In *Proceedings of ACM MobiCom 2009*, 2009.

[9] Y. Chen, Z. Li, and T. He. Twinbee: Reliable physical-layer cross-technology communication with symbol-level coding. In *Proceedings of IEEE INFOCOM 2018*, 2018.

[10] Z. Chi, Z. Huang, Y. Yao, T. Xie, H. Sun, and T. Zhu. Emf: Embedding multiple flows of information in existing traffic for concurrent communication among heterogeneous iot devices. In *Proceedings of IEEE INFOCOM 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*, 2016.

[12] Z. Chi, Y. Li, Y. Yao, and T. Zhu. Pmc: Parallel multi-protocol communication to heterogeneous iot radios within a single wifi channel. In *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, pages 1–10. IEEE, 2017.

[13] cisco.com. Cisco Global Cloud Index: Forecast and Methodology. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html, 2019.

[14] S. Clark. TazTag unveils first Android phone with NFC and ZigBee. https://www.nfcw.com/2012/02/26/313800/taztag-unveils-first-android-phone-with-nfc-and-zigbee/, 2012.

[15] eMarketer. US Adult Wearable Users and Penetration. https://www.emarketer.com/content/wearables-2019, 2018.

[16] P. Gawlowicz, A. Zubow, and A. Wolisz. Enabling cross-technology communication between lte unlicensed and wifi. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 144–152. IEEE, 2018.

[17] X. Guo, Y. He, X. Zheng, L. Yu, and O. Gnawali. Zigfi: Harnessing channel state information for cross-technology communication. In *Proceedings of IEEE INFOCOM 2018*, 2018.

[18] X. Guo, Y. He, X. Zheng, Z. Yu, and Y. Liu. Lego-fi: Transmitter-transparent ctc with cross-demapping. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2125–2133. IEEE, 2019.

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

[20] R. Hofmann, C. A. Boano, and K. Römer. X-burst: Enabling multi-platform cross-technology communication between constrained iot devices. In *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9, 2019.

[21] iPass. The Rise of Wi-Fi First. https://www.ipass.com/wp-content/uploads/2017/03/iPass-White-Paper-WiFi-First.pdf, 2017.

[22] W. Jiang, S. M. Kim, Z. Li, and T. He. Achieving receiver-side cross-technology communication with cross-decoding. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 639–652. ACM, 2018.

[23] W. Jiang, Z. Yin, S. M. Kim, and T. He. Transparent cross-technology communication over data traffic. In *Proceedings of IEEE INFOCOM, 2017*, 2017.

[24] W. Jiang, Z. Yin, R. Liu, Z. Li, S. M. Kim, and T. He. Bluebee: a 10,000 x faster cross-technology communication via phy emulation. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, page 3. ACM, 2017.

[25] S. J. Johnson. Introducing low-density parity-check codes. *University of Newcastle, Australia*, page V1, 2006.

[26] S. M. Kim and T. He. Freebee: Cross-technology communication via free side-channel. In *In Mobicom*, 2015.

[27] L. Kong and X. Liu. mzig: Enabling multi-packet reception in zigbee. In *Proceedings of the 21st annual international conference on mobile computing and networking*, pages 552–565. ACM, 2015.

[28] L. Li, Y. Chen, and Z. Li. Physical-layer cross-technology communication with narrow-band decoding. In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, pages 1–2. IEEE, 2019.

[29] Y. Li, Z. Chi, X. Liu, and T. Zhu. Chiron: Concurrent high throughput communication for iot devices. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 204–216. ACM, 2018.

[30] Z. Li and T. He. Webee: Physical-layer cross-technology communication via emulation. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 2–14. ACM, 2017.

[31] Z. Li and T. He. Longbee: Enabling long-range cross-technology communication. In *Proceedings of IEEE INFOCOM 2018*, 2018.

[32] 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*, pages 309–322. ACM, 2010.

[33] M. Schulz, D. Wegemer, and M. Hollick. Nexmon: The c-based firmware patching framework, 2017.

[34] Sears. 2.4GHz wireless digital video baby monitor. https://www.sears.com/xcsource-2.4-ghz-wireless-digital-video-baby/p-A026482260, 2020.

[35] Semtech. Using the SX1280/SX1281 in Low Power Applications. https://www.semtech.com/products/wireless-rf/24-ghz-transceivers/sx1280, 2019.

[36] Semtech. Semtech SX1280. https://www.semtech.com/products/wireless-rf/24-ghz-transceivers/sx1280, 2020.

[37] J. Shi, D. Mu, and M. Sha. Lorabee: Cross-technology communication from lora to zigbee via payload encoding. In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, pages 1–11. IEEE, 2019.

[38] TI. CC2650 SimpleLink™ 32-bit Arm Cortex-M3 multiprotocol 2.4 GHz wireless MCU. https://www.ti.com/product/CC2650, 2020.

[39] S. Wang, S. M. Kim, and T. He. Symbol-level cross-technology communication via payload encoding. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 500–510, July 2018.

[40] S. Wang, Z. Yin, Z. Li, and T. He. Networking support for physical-layer cross-technology communication. In *2018 IEEE 26th International Conference on Network Protocols (ICNP)*, pages 259–269, 2018.

[41] 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*, 2017.

[42] Z. Yin, Z. Li, S. M. Kim, and T. He. Explicit channel coordination via cross-technology communication. In *Proceedings of ACM MobiSys 2018*, 2018.

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

[44] X. Zheng, Y. He, and X. Guo. Stripcomm: Interference-resilient cross-technology communication in coexisting environments. In *Proceedings of IEEE INFOCOM 2018*, 2018.

[45] W. Zhou, T. Das, L. Chen, K. Srinivasan, and P. Sinha. Basic: backbone-assisted successive interference cancellation. In *MobiCom '16*, pages 149–161. ACM, 2016.