ZiSense: Towards Interference Resilient Duty Cycling in Wireless Sensor Networks

Xiaolong Zheng[1,2]
xiaolong@greenorbs.com

Zhichao Cao[2,3]
caozc@greenorbs.com

Jiliang Wang[2,3]
jiliang@greenorbs.com

Yuan He[2]
he@greenorbs.com

Yunhao Liu[2]
yunhao@greenorbs.com

[1] Department of Computer Science and Engineering, HKUST, Hong Kong
[2] School of Software and TNLIST, Tsinghua University, P. R. China
[3] WuXi Tsinghua IOT Center, WuXi, P. R. China

Abstract

To save energy, wireless sensor networks often run in a low duty cycle mode, where the radios of sensor nodes are scheduled between ON and OFF states. For nodes to communicate with each other, Low Power Listening (LPL) and Low Power Probing (LPP) are two types of rendezvous mechanisms. Nodes with LPL or LPP rely on signal strength or probe packets to detect potential transmissions, and then keep the radio-on for communications. Unfortunately, in many cases, signal strength and probe packets are susceptible to interference, resulting in undesirable radio on time when the signal strength of interference is above a threshold or a probe packet is interfered. To address the issue, we propose ZiSense, an energy efficient rendezvous mechanism which is resilient to interference. Instead of checking the signal strength or decoding the probe packets, ZiSense detects the existence of ZigBee transmissions and wakes up nodes accordingly. On sensor nodes with limited information and resource, we carefully study and extract short-term features purely from the time-domain RSSI sequence, and design a rule-based approach to efficiently identify the existence of ZigBee. We theoretically analyze the benefit of ZiSense in different environments and implement a prototype in TinyOS with TelosB motes. We examine ZiSense performance under controlled interference and office environments. The evaluation results show that, compared with state-of-the-art rendezvous mechanisms, ZiSense significantly reduces the energy consumption.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Network communications*

General Terms

Design, Experimentation, Performance

Keywords

Wireless Sensor Networks, Duty Cycling, Interference

Sensys'14, November 3–5, 2014, Memphis, TN, USA.

http://dx.doi.org/10.1145/2668332.2668334

1 Introduction

Due to the energy constraint on sensor nodes, it is of great importance to save energy and extend the network lifetime in wireless sensor networks. Recent studies show that radio activities are the main source of energy consumption [21]. Hence, a common approach to save energy is to make nodes working in a low duty cycle mode. The radio of a sensor node is scheduled between "sleep" (turn off the radio) and "wake up" (turn on the radio) state. Many existing sensor network applications show the significant improvement in energy efficiency brought by the asynchronous duty-cycled media access control (MAC) protocols [10] [22] [23], compared to the always-on methods. A crucial issue with those protocols is to design a rendezvous mechanism, which makes the sender and the receiver wake up during a same period of time to communicate with each other.

Low Power Listening (LPL) and Low Power Probe (LPP) are two well-known types of rendezvous mechanisms adopted in asynchronous duty-cycled MAC. BoX-MAC [22] and A-MAC [10] are state-of-the-art MAC protocols compatible with LPL and LPP. As shown in Fig. 1, in BoX-MAC, each receiver periodically wakes up to sample the energy level in the wireless channel, as is called, CCA (Clear Channel Assessment). If the energy level is above a predefined threshold, the receiver stays awake to receive the potential packet. The sender in BoX-MAC repeats transmitting a same data packet (called preamble [22]) until an ACK (acknowledgement packet from the receiver) is received. In A-MAC, each receiver periodically wakes up to send a probe. Instead of transmitting the preamble, the sender meets the receiver by successfully decoding the probe from the intended receiver.

Both LPL and LPP can significantly reduce the energy consumption in low duty cycling mode. Nevertheless, they

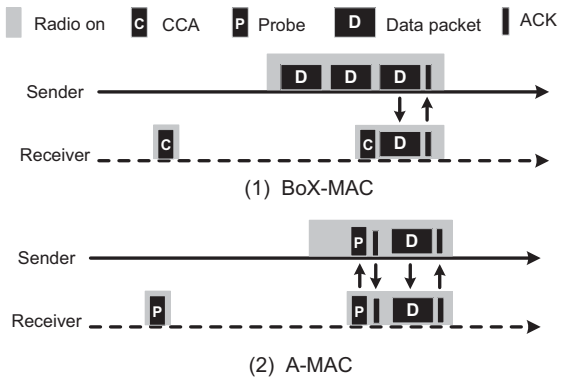


Figure 1. The duty cycled communication flow of BoX-MAC and A-MAC

suffer performance degradation in noisy environments with signal interference [21]. Specifically, a node under LPL is likely to incorrectly regard a non-ZigBee interfering signal as an interested signal and improperly keeps the radio on (false wake-up), leading to considerable but unnecessary energy consumption. The probability of false wake-up even significantly goes up in the crowded unlicensed 2.4GHz ISM band [21], which is used by various wireless technologies such as ZigBee [17], WiFi [19], Bluetooth [18], and microwave ovens. Similar problems also exist with LPP. If a probe from the receiver is corrupted by interference, the sender will stay awake and wait for the intended probe for a long period of time (idle listening). In short, simply relying on the signal strength to maintain a rendezvous mechanism suffers the false wake-up problem. On the other aspect, relying on successfully decoded packets suffers the idle listening problem. The ubiquitous cross technology interference seriously degrades the efficacy and efficiency of the existing rendezvous mechanisms.

To address the above issue, we propose ZiSense, an energy efficient rendezvous mechanism tailored to sender-initiated MAC protocols in noisy environments. Instead of checking signal strength or decoding the probe, ZiSense detects potential ZigBee transmissions according to the featured patterns of ZigBee signals. Nodes in ZiSense wake up only when ZigBee signals are detected.

The design of ZiSense face several non-trivial challenges in practice. First, the available information on most ZigBee-compatible devices is very limited (e.g., only RSSI on CC2420 radio). Second, the computation resource is very limited. Third, the time used to obtain the ZigBee information should be short enough to restrict the overhead. To address these challenges, we first carefully select features from time domain RSSI samples to effectively distinguish ZigBee signals from other interfering ones. We reduce the sampling time of the features and improve the RSSI sampling technique to minimize the sampling overhead. Compared to LPL, our sampling method does not incur extra overhead. Last but not least, we design a light-weight rule-based approach to distinguish ZigBee signals from interferences.

The main contributions of this work are summarized as follows.

- We empirically study the performance of the existing rendezvous mechanisms and disclose that energy detection in LPL is too simple to filter the interference and probe decoding in LPP is too strict to cope with the interference, which are the fundamental problems in those mechanisms.
- We propose ZiSense, an energy efficient rendezvous mechanism that leverages RSSI sequence pattern to recognize ZigBee signals to avoid unnecessary wake-ups under interference environments. We theoretically validate the performance gain and analyze the overhead of ZiSense under different environments to show ZiSense is resilient to interference.
- We implement ZiSense with TinyOS and TelosB motes, and evaluate its performance in different environments. The results show that ZiSense significantly reduces the energy consumption of sensor nodes under interference, compared to the existing rendezvous mechanism.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 illuminates the motivations of this work. In Section 4, we empirically study the short-term characteristics of RSSI sequences of different 2.4GHz signals. Section 5 presents an overview of ZiSense and Section 6 elaborates on the identification algorithm design. In Section 7, we analyze the performance of the representative rendezvous mechanisms. Section 8 presents the implementation details. Section 9 shows the evaluation results of ZiSense in both controlled and real environments. We conclude this work in Section 10.

2 Related Work

Interference aware rendezvous mechanism. Most of the existing LPL compatible approaches adopt a fixed CCA threshold to detect the ZigBee transmissions. They suffer from serious false wake-up problem [22] [23]. The authors in [28] propose AEDP that adaptively adjusts the CCA threshold to alleviate this problem. However, AEDP may still have false wake-ups when the signal strength of interference is higher than the ZigBee. The scenario should not be uncommon in current environment with cross technology interference [21] [33]. In ZiSense, nodes keep awake by recognizing the ZigBee signal according to its time domain features, which is irrelevant with the signal strength. Therefore, ZiSense is more adaptive for the general interference situations.

Coexistence. Recently, many works study the coexistence between ZigBee and other interference to enhance the robustness of ZigBee transmissions. The authors in [21] measure the impact of 802.11 interference on 802.15.4 networks and propose to use redundant headers and the forward error correction code to alleviate packet corruption. The authors in [32] study the chip error patterns under interferences. The impact of 802.11 interference on body sensor networks is studied in [12]. The authors also find that bit errors in 802.15.4 packets are temporally correlated with 802.11 traffic. Based on this correlation, they further propose an error recovery method that mitigates the effect of interference [13]. The authors in [16] propose an approach to enable ZigBee perform transmissions during the whitespace of

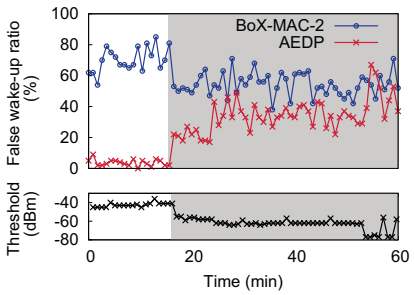


Figure 2. False wake-up ratio of threshold-based CCA checking in an office environment

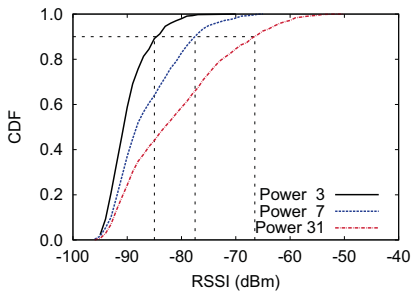


Figure 3. CDF of RSSI on Mirage testbed with different RF powers

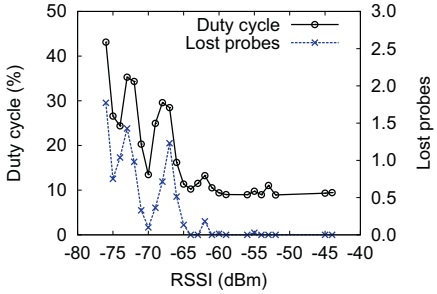


Figure 4. Duty cycle and the number of probe loss of A-MAC under various link qualities

WiFi traffic. These works concentrate on mitigating the effect of interference on receiving and decoding packets. In this paper, we study energy inefficiency of the rendezvous mechanism incurred by interference in low duty cycle media access. Our work is complementary to above works since we reduce energy consumption from a different aspect.

Interference classification. Many efforts have been made to interference classification. Airshark [25] and WiFiNet [26] leverage powerful WiFi hardware to get the spectrum information to detect and classify non-WiFi interference. DOF [15] provides the local wireless information plane, including the information of the interferers. It is proposed in [7] to scan 16 ZigBee channels to get the spectrum characteristics for classification. The authors in [4] design a framework to scan the 2.4GHz band. ZiFi [33] and ZiFind [11] recognize WiFi signal by detecting periodical beacons in WiFi. They depend on a relative long-term sampling since the default period of WiFi beacon is 100ms. SoNIC [14] proposes a method to classify non-ZigBee interference by the observation that different interference will result in different corruption patterns on received packets. SoNIC needs to identify the corrupted bits and then extracts the features of the signal corresponding to the corrupted bits.

These methods aim at providing a detailed classification for non-ZigBee interference. They either rely on dedicated hardware or complicated algorithm together with long-term sampling. None of them provides a light weight feature fetching and identification algorithm to fulfill the needs of the short-term ZigBee signal detection. Different from above methods, in ZiSense, we develop a novel feature extraction and identification algorithm to sense the existence of ZigBee in a short time, with only RSSI information in time domain.

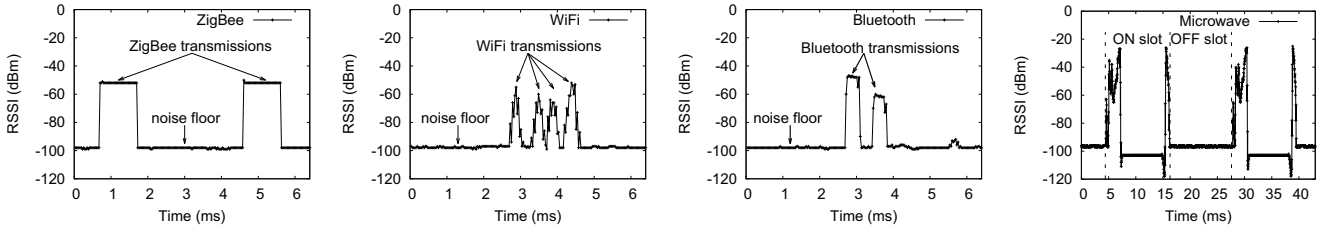
3 Motivation

3.1 Impact of Interference on LPL

To study the impact of interference on existing LPL mechanism, we conduct a series of experiments. We deployed three TelosB motes in an office environment on channel 22. Node 1 acts as a sender and broadcasts a packet every 10 seconds; node 2 acts as a receiver and runs BoX-MAC-2 [22], the default LPL MAC in TinyOS; node 3 acts as another receiver and runs AEDP [28], the adaptive-threshold method. The sleep interval of both receivers is set to 512ms. We measure the number of false wake-ups, i.e. the number of wake-ups without receiving any data. Then we calculate the false wake-up ratio as the number of false wake-ups to the total number of wake-ups. During the experiment, we vary the position of the sender to change the link qualities to the receivers. We switch the positions of two receivers and repeat the experiment to exclude the influence of spatial differences. The repeated experiment shows similar results. Hence, we omit its results and only show the results of one experiment.

Fig. 2 plots the false wake-up ratios for the two receivers with two different protocols. First, we can see that BoX-MAC-2 experiences a very high false wake-up ratio. In most of the time, the false wake-up ratio is higher than 40%. Further, we evaluate the performance of AEDP with an adaptive threshold. As shown in Fig. 2, AEDP can achieve a low false wake-up ratio in the beginning. We check the beginning region and find that the RSSI of link, which varies in $[-50dBm, -36dBm]$, is higher than the interference. AEDP can therefore filter the interference from the communication link and effectively reduce the false wake-ups to achieve a low duty cycle. However, as the link RSSI decreases, the performance of AEDP degrades. When the link RSSI is between $[-77dBm, -55dBm]$, as shown in the grey region in Fig. 2, the false wake-up ratio becomes very high. We investigate the data and find that during the grey region, the interested signal and the interference cannot be separated based on a signal strength threshold. Thus AEDP cannot find an appropriate threshold to avoid false wake-ups.

AEDP requires that the link RSSI is higher than the signal strength of interference to work effectively under interference. However, high RSSI is not always common for real low power wireless links. For example, we use SING [29] dataset to examine the link RSSI in a real indoor system. Fig. 3 shows that 90% of links have a RSSI lower than $-66dBm$, even though the highest transmitting power (0dBm) is adopted. Other works and experiments also show similar results of the link RSSI [30]. Considering the interference sources usually have a higher transmission power (e.g., WiFi), it is not uncommon that the signal strength of interference is higher than link RSSI [21]. The signal of interference can mask the ZigBee signal, making it difficult for AEDP to distinguish those two types of signals.



(a) ZigBee (b) WiFi (c) Bluetooth (d) Microwave oven

Figure 5. RSSI patterns of different 2.4GHz technologies

3.2 Impact of Interference on LPP

We also conduct experiments to show the impact of interference on LPP performance. In LPP, a sender listens to the channel for probe packets to learn the existence of the interested receiver. Upon receiving a probe from the intended receiver, the sender will send its packets. However, the probe packets may get lost under interference, resulting in idle listening on the sender. We use two nodes with A-MAC [10], the most recent protocol with LPP. In the experiment, the receiver sends a probe every 512ms and the sender generates a packet every 2 seconds. Fig. 4 presents the duty cycle of the sender with A-MAC under different environments. Similar to the results in LPL, when the link RSSI is high, (e.g., between $[-60dBm, -40dBm]$), the probe packets rarely lost and the duty cycle ratio is low. When the link RSSI becomes lower, the probe packets are not reliable. Once a probe packet is lost, the sender needs to wait for an entire sleep interval (i.e. 512ms) without sending any packet. However, corruption of probe packets does not necessarily means the data packet is also corrupted since the repeated transmissions of data packets can sufficiently increase the receiving probability. The fragile single probe packet in existing LPP wastes the transmission chances, resulting in idle listening at the sender. This idle listening time significantly increases the duty cycle.

3.3 Summary

To summarize, the performances of both LPL and LPP are degraded significantly in presence of interference. The fundamental problem is that LPL and LPP rely on either checking the signal strength or decoding the probe packets for a rendezvous of the sender and receiver. Both of these techniques are susceptible to interference. Checking the signal strength is a loose condition that allows too much interference to wake up nodes. In contrast, decoding probe packets is a stringent condition that makes the senders ignore some transmission chances, resulting in idle listening. Our key insight is that a node should only wake up when there is a ZigBee transmission rather than a detected high energy or a decoded probe packet.

4 Short-term RSSI Characteristic Study

To wake up nodes only when ZigBee transmissions exist, it should be able to identify ZigBee from the various coexisting wireless techniques, with limited information and resource available on sensor nodes in a short time period. Hence, we focus on the question: can ZigBee transmission be *effectively* and *efficiently* identified with common sensor nodes? In this section, we study the short time patterns of the RSSI sequences during ZigBee transmissions. We also study some other common wireless techniques in 2.4GHz band such as WiFi, Bluetooth and microwave ovens which are usually considered in studies of interference classification [14] and co-existence [13].

4.1 Characterizing Common Technologies

We measure the RSSI sequences of common technologies in 2.4GHz band under the controlled environments to obtain their accurate characteristics. To achieve the fine-grained RSSI information, we reimplement several interfaces of CC2420 in TinyOS to increase the sampling rate to 31.25KHz, i.e., 32μs/sample. Fig. 5 presents the RSSI sequences of different technologies sampled by a TelosB mote. Noise floor is the received signal strength of the background noise when there is no wireless activity on the channel.

ZigBee. Fig. 5(a) presents the sampled RSSI sequence during ZigBee transmissions, showing flat high RSSI segments. The modulation technique specified by the underlying IEEE standard 802.15.4 [17] is Direct Sequence Spread Spectrum (DSSS). The length of a CC2420 packet is [18,133]bytes [1]. It determines that the range of valid on-air time is $[576, 4256]\mu s$ under the standard rate of 250kbit/s.

WiFi. Fig. 5(b) shows the RSSI sequence during four 802.11n packet transmissions from a 802.11b/g/n Access Point (AP). At first glance, we can see that there are four spikes in the sequence, each of which corresponding to a packet. Fluctuations are observed even during one packet's transmission. This is due to Orthogonal Frequency Division Multiplexing (OFDM), the multiple sub-carriers modulation technique adopted by WiFi [19]. In OFDM, each subcarrier has a certain level variation of signal strength. The received signal is a sum of the signals on all the orthogonal sub-carriers. Thus the variation of the sum will be larger than that of a single carrier. Hence, spurious high power peaks occur when signals from different sub-carriers are added constructively [27]. The valid packet lengths and data rates specified by the underlying IEEE standard 802.11 [19] limits the on-air time of a WiFi packet in $[192, 542]\mu s$, which is shorter than the minimum ZigBee on-air time.

Bluetooth. As shown in Fig. 5(c), the high RSSI subsequence during Bluetooth transmissions are flat as ZigBee since its underlying IEEE standard 802.15.1 [18] specifies Frequency-hopping spread spectrum (FHSS) as the modulation technique, which is also a single-carrier modulation technique. Nevertheless, Bluetooth adopts a frequency hopping technique. The standard hopping rate is 1600 hop/s,

Table 1. Characteristics of common 2.4GHz technologies

Wireless technology	On-air time	MPI	PAPR	UNF
ZigBee	[576, 4256]μs	2.8ms or 192μs	≤ 1.3	FALSE
WiFi	[194, 542]μs	$\geq 28\mu$s	≥ 1.9	FALSE
Bluetooth	366μs	NA	≤ 1.3	FALSE
MWO	10ms	10ms	≥ 2.9	TRUE

i.e., 625μs residence time in one channel. The standard also specifies the transmission time in one channel is 366μs, resulting in shorter on-air time than a ZigBee transmission.

Microwave oven (MWO). Although the microwave oven is covered by a Faraday cage, leaked energy can still interfere ZigBee transmissions. Fig. 5(d) shows a distinct RSSI sequence pattern of MWOs. The RSSI sequence during a microwave oven operating show fluctuations up and down on the noise floor while other technologies have only RSSI samples above the noise floor. This is due to the saturation of the intermediate frequency amplifier chain. Similar phenomenon is also observed in other studies [5]. The residential microwave ovens work in ON-OFF mode, in synchronism with the frequency of alternating current supply. Therefore, the on-air time and the interval between signals are decided by the working modes of microwave ovens and the frequency of power supply.

4.2 Feasibility of Using the Short-term RSSI Sequence for Identifying ZigBee

The RSSI sequences of different wireless technologies exhibit different patterns. A following question is: can we efficiently distinguish those patterns based on only RSSI information within a short time? To answer such a question, we investigate the features of ZigBee transmissions as well as some other common wireless technologies in 2.4GHz. We find there indeed exist several features that can be leveraged to identify ZigBee transmission based on the short-term RSSI information, as listed in Table 1.

On-air time. Under the data rate of 250Kbps, specified by the standard of ZigBee, the on-air time of a normal ZigBee data packet in TinyOS is between $[576\mu s, 4256\mu s]$. Unlike ZigBee, WiFi and Bluetooth have a shorter on-air time, while microwave ovens have a longer on-air time.

MPI. Minimum Packet Interval (MPI) is the minimum interval between successive transmissions. ZigBee's MPI is defined as the interval between two successive preambles. The MPI of adjacent unicast packets is 10ms by default settings in TinyOS-2.1.2, and is further reduced to 2.8ms in AEDP [28]. The MPI between adjacent broadcast packets is 192μs by default settings of CC2420 [20]. WiFi waits for at least a DIFS time before the next transmission, which is 28μs for 802.11 g/n and 50μs for 802.11b. MPI of Bluetooth refers to the interval between successive packets on the same channel with ZigBee since Bluetooth follows a pseudorandom hopping. The MPI of MWOs is defined as time between two ON slots, which equals the time of an OFF slot.

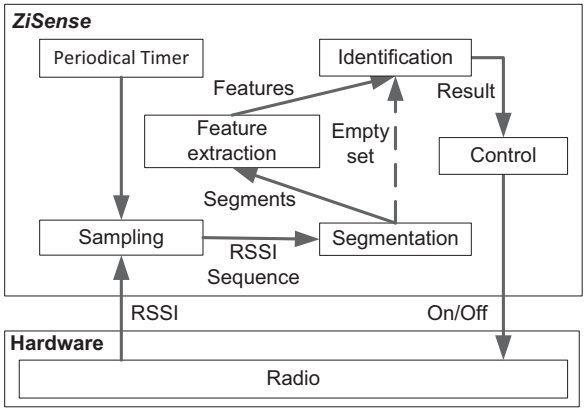


Figure 6. Framework of ZiSense

PAPR. Due to different PHY modulation techniques, different wireless technologies experience different received energy fluctuations. Peak to Average Power Ratio (PAPR) is a common measure of the fluctuation of signal power. We apply PAPR to analyze the fluctuations of RSSI sequences. As previous studies [27] have shown, 802.11g/n have a large PAPR (≥ 1.9). MWOs also have a large PAPR. In contrast, ZigBee and Bluetooth have a relatively small PAPR (≤ 1.3) because they employ the single-carrier modulation techniques.

UNF. Under Noise Floor (UNF) is an indicator, describing whether a segment contains RSSI lower than the minimum possible noise floor. The normal range of RSSI returned by CC2420 is [-100,0]dBm, implying a minimum possible noise floor of -100dBm. However, we found that the RSSI can be lower than -100dBm during the ON phase of microwave ovens. This is an exclusive feature of MWO. If there is a sample with RSSI smaller than the minimum possible noise floor, UNF is $TRUE$; otherwise, it is $FALSE$.

4.3 Summary

There indeed exist several features that can be leveraged to *effectively* identify ZigBee transmissions. These features are extracted from network standards, hardware specifications and modulation methods etc. Meanwhile, most features can be extracted in a very short time period as observed in aforementioned measurements. For example, the longest duration of these features appears about several milliseconds, as shown in Table 1. This shows that leveraging those features can *efficiently* identify ZigBee transmissions without incurring additional sampling overhead comparing to existing sampling based LPL and decoding based LPP mechanisms.

5 ZiSense Overview

The empirical results in Section 4 have demonstrated the feasibility of identifying ZigBee with only RSSI information within a short period. This paves the way for the design of a more interference-resilient rendezvous mechanism, which wakes up nodes only in presence of ZigBee transmissions. In this section, we present ZiSense, a ZigBee sensing based rendezvous mechanism tailored to the sender-initiated duty-cycled MAC protocols.

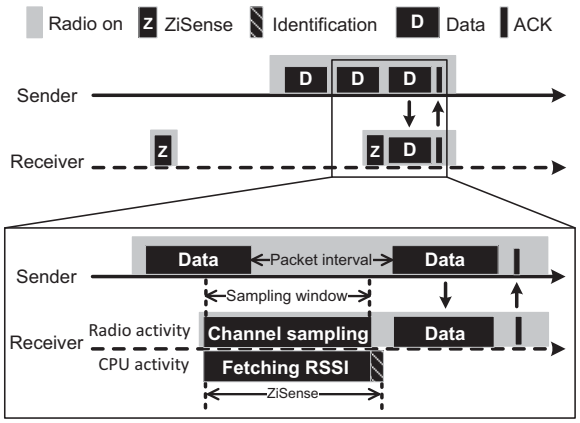


Figure 7. The duty cycled communication flow of ZiSense compatible MAC

5.1 Working Flow

Fig. 6 presents the framework of ZiSense. The first component is the RSSI sampling which obtains RSSI samples from the radio and provides the RSSI sequence to the segmentation component. The segmentation component analyzes the input sequence to obtain a set of segments. We define a segment as a sub-sequence of the obtained RSSI sequence, which is composed of consecutive samples with RSSI readings different from the noise floor. Therefore, if no segment is found, ZiSense consider the channel as idle and no potential packet exists. The radio will be directly turned off. Otherwise, the feature extraction component takes the detected segments as input and calculates the features for each segment (as listed in Table 1). The extracted features of a segment are represented in the form of feature vector, $\mathbf{f} = (PAPR, T_{on}, MPI, UNF)$. Then the obtained feature vectors will be provided to ZigBee identification component to determine whether a ZigBee segment exists. If yes, ZiSense keep the radio on for the potential packets. Otherwise, the radio will be turned off.

In a sender-initiated MAC protocol integrated with ZiSense, the sender initiates the transmission and repeats transmitting the data packets (called preamble) until an ACK is received. As shown in Fig. 7, the receiver with ZiSense periodically wakes up to perform channel sampling and the operations in Fig. 6. If a ZigBee signal is sensed, the node will turn on the radio to receive the potential packets. Otherwise, it turns off the radio. In this way, comparing with LPL, ZiSense avoids the energy wasted on false wake-ups by accurate ZigBee signal identification. In ZiSense, since the receiver might overhear multiple preamble packets after it wakes up, packet transmission is more resilient to interference than LPP [6], in which the probe is only transmitted once. In section 9.5, we investigate the influences of interference on packet retransmissions for both ZiSense and LPP.

5.2 Sampling and Segmentation

To detect the existence of possible transmission in LPL, the sampling window must be longer than the minimal packet interval from the sender. As shown in Fig. 7, the sender transmits adjacent packets with an interval for receiving the potential ACK since there is no need to repeat the transmission if the receiver has received the packet. Therefore, the waiting time should be at least longer than the ACK delay, T_{ACK}. The sampled RSSI sequence with W samples will be fed into segmentation component for further processing.

Segmentation aims at extract useful information from the RSSI sequence. Therefore, the segmentation component outputs segments, each consists of consecutive RSSI samples. Since an effective signal usually results in a sudden difference to the noise floor, ZiSense adopts a single threshold method to detect the start and end points of each segment. If the difference between the RSSI and the noise floor (denote as $Noise$) is larger than a threshold th_d, ZiSense detects the start of a segment. Similarly, ZiSense detects the end of a segment when the difference falls below th_d.

Denote the RSSI sequence collected in sampling as: $X = \{x_1, x_2, ..., x_W\}$. Then, the sets of start (S) and end (E) positions of segments are:

$$S = \{s \big| |x_{s-1} - Noise| < th_d, |x_s - Noise| \geq th_d\} \quad (1)$$

$$E = \{e \big| |x_e - Noise| \geq th_d, |x_{e+1} - Noise| < th_d\} \quad (2)$$

We sort S and E in ascending order and put them in two separated arrays I_S and I_E. Then the k-th segment can be represented by $X_k = \{x_{I_S(k)}, x_{I_S(k)+1}, ..., x_{I_E(k)}\}$. To eliminate the impact that the start position or end position is out of the sampling window, we add two samples with RSSI equal to noise floor at the start and end of the sampling sequence before segmentation. Hence, we have $|I_S| = |I_E| = K$.

5.3 Feature Extraction

Feature extraction component takes the set of extracted segments as input and calculates the values of features listed in Table 1. Then each segment will have a feature vector $\mathbf{f} = (PAPR, T_{on}, MPI, UNF)$. The set of feature vectors of all segments is then provided to the ZigBee identification algorithm.

On-air time. Based on Eq. 1 and Eq. 2, on-air time of segment k can be calculated as:

$$T_{on}(k) = (I_E(k) - I_S(k)) \cdot T_s \quad (3)$$

where T_s is the sampling period.

PAPR. The RSSI samples must be first normalized. Denote the normalized RSSI sequence as $X' = \{x'_1, x'_2, ..., x'_W\}$, where $x' \in [0,1]$. The PAPR of segment k can be calculated as:

$$PAPR(k) = \frac{\max\{x'^2_l | I_S(k) \leq l \leq I_E(k)\}}{\overline{X'^2_k}} \quad (4)$$

where $\overline{X'^2_k}$ denotes the average of the squared values of the elements in segment X'_k.

MPI. To calculate the MPI, we need to identify the segments belong to the same signal source. We observe that the average RSSI and on-air time of segment from the same signal source do not vary significantly during a short sampling period. Hence, we determine the segments belonging to the same signal source by the similar average RSSI and on-air

time. Given a segment k, the nearest segment from the same signal source can be determined as

$$\begin{aligned} j &= \arg\min_j |k-j| \\ \text{subject to:} \quad & \left|T_{on}(k)-T_{on}(j)\right| \le \delta, \\ & \left|\overline{X_k}-\overline{X_j}\right| \le \varepsilon, 1 \le j \ne k \le K \end{aligned} \quad (5)$$

where $\overline{X_k}$ and $\overline{X_j}$ are the average RSSI readings of segment k and j, and δ and ε are error thresholds for deciding same on-air time and average RSSI readings, respectively. Then MPI of segment k is calculated as:

$$MPI(k) = \left(I_S(max\{k,j\}) - I_E(min\{k,j\})\right) \cdot T_s \quad (6)$$

where T_s is the sampling period. It is possible that MPI cannot be calculated when only one segment from a signal source is found in the sampling window. Under such a case, to ensure ZigBee transmission is not ignored, ZiSense makes conservative decisions by filling in a valid MPI complying with ZigBee.

UNF. $UNF(k)$ is the Boolean indicator, indicating whether segment k has the RSSI values lower than the minimum possible noise floor, th_n. It can be decided as follows.

$$UNF(k) = \begin{cases} TRUE, & \exists x_i, i \in [I_S(k), I_E(k)], x_i < th_n; \\ FALSE, & Otherwise. \end{cases} \quad (7)$$

After extracting the features, segment k will have a feature vector $\mathbf{f_k}$ with value $(PAPR(k), T_{on}(k), MPI(k), UNF(k))$. Then the set of feature vectors is provided to identification algorithm to determine whether ZigBee exists.

6 ZigBee Identification

Given a set of feature vectors, ZigBee identification component determines whether any feature vector matches the characteristic of a valid ZigBee segment. In this section, we explain our design of the identification algorithm. We first propose a set of deterministic rules based on ZigBee's underlying standard, IEEE 802.15.4. But we find the features may be corrupted in practice, resulting in the mismatch of a valid ZigBee segment. We then propose an improved algorithm which is able to identify the ZigBee segments even with some corrupted features to enhance the robustness of ZiSense. We finally compare the accuracy of different methods and discuss their application scenarios.

6.1 Rule Generation

ZigBee signal is differentiable from other co-existing wireless technologies by checking whether all the features meet the ZigBee standard. On this basis, we propose a set of deterministic rules to identify ZigBee, as shown in Algorithm 1.

To determine a segment is ZigBee or not, Algorithm 1 has four conditions to check: (1) $C1 : PAPR \le PAPR_{ZigBee}$; (2) $C2 : T_{on} \ge T_{min}$; (3) $C3 : |MPI - MPI_{valid}| \le \delta$; (4) $C4 : UNF = FALSE$. Since the sampling window is shorter than the maximum on-air time of a valid ZigBee packet (denote as T_{max}), the extracted T_{on} is impossible to be larger than T_{max}. Therefore, Algorithm 1 does not check whether $T_{on} > T_{max}$. Since MPI_{valid} has two values, 2.8ms for unicast and 192μs for broadcast, then $C3$ considered as being satisfied as long as one of the MPI_{valid} values make the condition satisfied. Algorithm 1 is a strict testing that filters out all the invalid segments with any violation to the conditions. Only the ZigBee segments with the correct feature vector can pass all the checking conditions. Hence, the condition vector of a ZigBee segment $\mathbf{C} = (C1, C2, C3, C4)$ should be (T, T, T, T), where T is $TRUE$ and F is $FALSE$.

Algorithm 1: *Deterministic rules to identify ZigBee*

Input : feature vector $\mathbf{v} = (PAPR, T_{on}, MPI, UNF)$;
Output: whether the segment is ZigBee or not.

1 **if** $PAPR > PAPR_{ZigBee}$ **then return** FALSE;
2 **else if** $T_{on} < T_{min}$ **then return** FALSE;
3 **else if** $|MPI - MPI_{valid}| > \delta$ **then return** FALSE;
4 **else if** $UNF = TRUE$ **then return** FALSE;
5 **else return** TRUE;

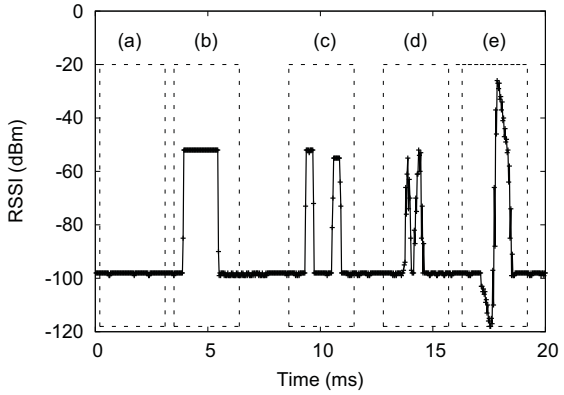


Figure 8. The illustration of how ZiSense recognizes ZigBee when no feature is corrupted

To show how Algorithm 1 works, we present an example trace with segments from different wireless technologies in Fig. 8. The dashed rectangles are sampling windows.

(a) A flat segment with RSSI around the noise floor. Actually, in this case, the output of segmentation is an empty set. ZiSense decides there is no ZigBee.

(b) A flat segment with RSSI above the noise floor. ZiSense decides ZigBee exists due to the valid condition vector (T, T, T, T).

(c) Two flat segments with RSSI above the noise floor. Even PAPR is small, the on-air time is too short to be ZigBee ($T_{on} < T_{min}$ violates $C2$). Thus Algorithm 1 decides there is no ZigBee signal.

(d) Two fluctuant segments with RSSI above the noise floor. $C1$ and $C3$ are violated. Hence, Algorithm 1 decides there is no ZigBee signal.

(e) A fluctuant segment with RSSI both above and below the noise floor. Algorithm 1 decides it is not ZigBee signal since the $C1$ and $C4$ are violated.

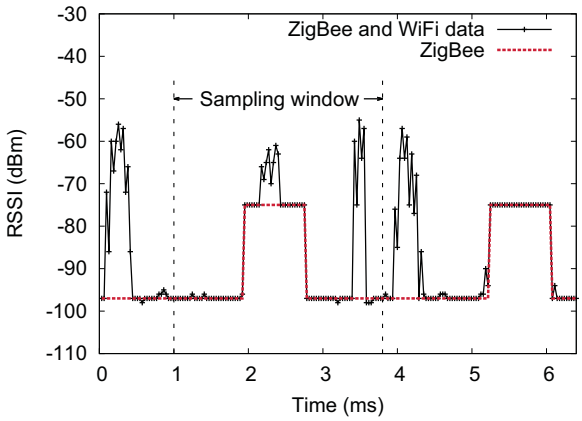


Figure 9. Example of segments with corrupted *PAPR*

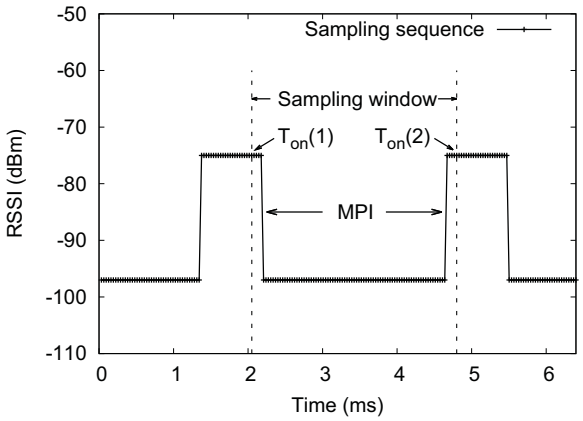


Figure 10. Example of ZigBee segments with $T_{on} < T_{min}$

Table 2. Condition vectors of different technologies

Signal source	Condition vector C	# of violations
ZigBee	(T,T,T,T)	0
WiFi	(F,F,F,T)	3
Bluetooth	(T,F,F,T)	2
MWO	(F,T,F,F)	3

We deduce the condition vectors of some other common wireless technologies in Table 2, from their corresponding standards. We also calculate the number of violated conditions of different technologies. We can find that the ZigBee condition vector is not only distinguishable but also has a larger number of conditions with values different with other technologies, demonstrating the effectiveness of deterministic rules to identify ZigBee.

6.2 Handle Corrupted Features

In Algorithm 1, if one of the conditions is violated, the segment will be treated as non-ZigBee. As a consequence, a ZigBee segment with corrupted features will be treated as a non-ZigBee signal incorrectly. This is undesirable for ZiSense's design since ZiSense follows *a conservative design principle* that tries to identify ZigBee transmissions as much as possible. Therefore, we discuss several possible cases with corrupted features and enhance Algorithm 1 to improve the robustness.

Case 1: *PAPR* is corrupted due to the overlapped concurrent signals. When the wireless environment is very crowded, the interference signal may overlap with a ZigBee signal. Then *PAPR* of the segment generated by this overlapped ZigBee signal may be corrupted, leading to the condition vector $\mathbf{C_{E1}} = (F,T,T,T)$. An example is shown in Fig. 9.

In this case, we regard the segment with $\mathbf{C_{E1}}$ as ZigBee. This extension is safe due to following two reasons. First, a ZigBee segment with $\mathbf{C_{E1}}$ is impossible to be regarded as other technologies since none of them has a condition vector with value (F,T,T,T). Second, other technologies are hard to be identified as ZigBee. This is because to have a condition vector as $\mathbf{C_{E1}}$, at least two conditions of the non-ZigBee segments have to be violated.

Table 3. Condition vectors of different technologies

Signal source	Condition vector C	# of violations
ZigBee	(T,T,T,T) (F,T,T,T) (T,F,T,T)	0
WiFi	(F,F,F,T)	2
Bluetooth	(T,F,F,T)	1
MWO	(F,T,F,F)	2

Case 2: T_{on} can be corrupted if channel sampling begins at the packet tail and ends at the packet head, as shown in Fig. 10. Since the ZigBee transmissions are only partly detected, the extracted segments will have a shorter on-air time than expected. But the *MPI* in such a case usually can be correctly extracted. As a result, a ZigBee segment in this case will have the condition vector $\mathbf{C_{E2}} = (T,F,T,T)$. Therefore, we extend our algorithm to regard a segment with $\mathbf{C_{E2}}$ as ZigBee.

This extension is safe from the perspective of identifying ZigBee segments since $\mathbf{C_{E2}}$ is still different with the condition vectors of other co-existing technologies. However, the extension increases the probability of regarding interference as ZigBee. Among the interference sources, the Bluetooth with condition vector (T,F,F,T) is closest to $\mathbf{C_{E2}}$. The frequency hopping adopted in Bluetooth makes it possible for a Bluetooth device to jump back to the same channel after exact $MPI_{valid} \pm \delta$ time, resulting in a condition vector (T,F,T,T) and then false wake-ups. We argue that the probability that a Bluetooth segment has a *MPI* exactly equal to MPI_{valid} is low. This is also verified in our office environment experiments in Section 6.3.

Other cases: $C3$ and $C4$ are possible to be corrupted but with a low probability. $C3$ of a ZigBee segment is hard to be violated since *MPI* is corrupted only when a segment from other technology has the same average RSSI and on-air time as the valid ZigBee segments. $C4$ of a ZigBee segment is impossible to be violated since $UNF = TRUE$ is an exclusive feature of MWOs. To handle more complicated cases, ZiSense can extend the sampling period to precisely cap-

Algorithm 2: *Robust ZigBee identification*

Input : feature vector $\mathbf{v} = (PAPR, T_{on}, MPI, UNF)$;
Output: whether the segment is ZigBee or not.

```
1 if PAPR ≤ PAPR_ZigBee then
2     if (UNF = FALSE and |MPI − MPI_valid| ≤ δ) then
          return TRUE;
3     else
          return FALSE;
4 else
5     if T_on < T_min then return FALSE;
6     else if |MPI − MPI_valid| > δ then return FALSE;
7     else if UNF = TRUE then return FALSE;
8     else return TRUE;
```

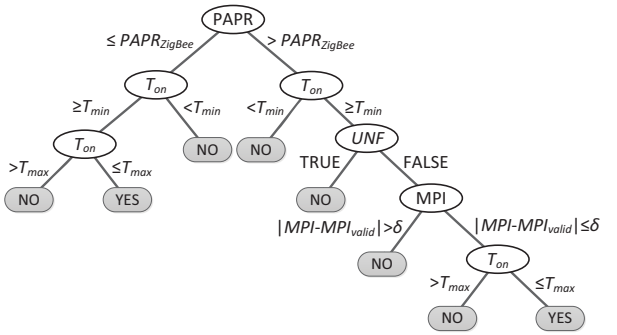


Figure 11. The trained decision tree according to C4.5

ture the correct features when too many false wake-ups occur. However, this extension increases the baseline energy. Hence, in current design, we do not extend ZiSense further to handle more complicated cases for the low baseline energy.

Based on the aforementioned enhancements, ZiSense enlarges the set of valid condition vectors of ZigBee to improve the robustness, as shown in Table 3. According to the new set of valid condition vectors, ZiSense adopts a more robust algorithm to identify ZigBee transmissions, as shown in Algorithm 2.

6.3 Evaluation of the Identification Algorithm

To validate the effectiveness of Algorithm 2 in practice, we collect 11621 labeled segments under a controlled office environment in presence of WiFi, Bluetooth and MWOs. We use a pair of TelosB motes into the environment to perform transmissions. The sender keeps sending packets with various packet sizes. The receiver collects the channel samplings. The sender and receiver are synchronized before collecting the traces. To obtain the ground truth, we label a segment as ZigBee if the sender sends a ZigBee packet at the same time. Otherwise, we label the segment as non-ZigBee.

Then the algorithms take the labeled traces as input to validate the effectiveness of our algorithms. For comparison, we also directly take the labeled segments as the training set to train a decision tree according to C4.5 algorithm [9]. The resulted decision tree is shown in Fig. 11. We adopt a 10-fold cross validation to obtain the accuracy of decision tree. The identification accuracies of different algorithms are presented in Table 4. False negative (FN) rate is the probability that a ZigBee signal is detected as a non-ZigBee signal, leading to retransmissions at the sender in the sender-initiated MAC protocols. False positive (FP) rate is the probability that a non-ZigBee signal is detected as a ZigBee signal, resulting in false wake-ups. True positive (TP) rate is the probability that a ZigBee signal is successfully detected. True negative (TN) rate is the probability that a non-ZigBee signal is correctly detected.

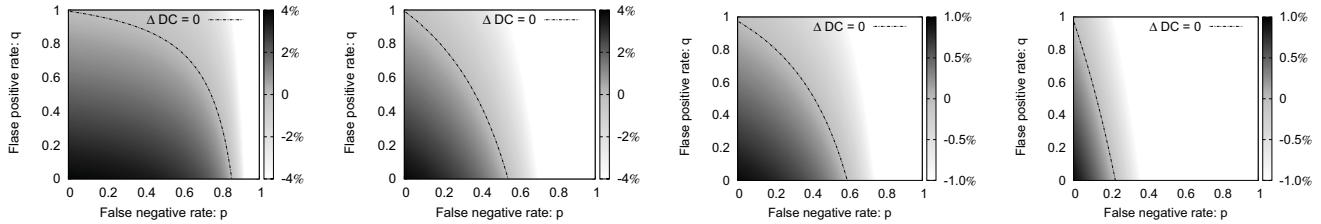
Table 4. Identification accuracies of different algorithm

Algorithm	TP rate	FN rate	TN rate	FP rate
Algorithm 1	87.6%	12.4%	100%	0%
Algorithm 2	97.5%	2.5%	97.6%	2.4%
C4.5	97.3%	2.7%	99.1%	0.9%

Algorithm 1 can only identify 87.6% ZigBee segments. This is because the corrupted ZigBee segments fail passing the condition checking. Algorithm 1 does not consider any non-ZigBee as ZigBee, resulting in 0% false wake-up. Algorithm 2 obviously improves the TP rate and is able to identify 97.5% ZigBee segments correctly. However, it also increases the FP rate to 2.4%. The decision tree directly trained from traces achieves 97.3% TN rate and 0.9% FP rate. Compared to Algorithm 2, the TP rate is decreased by 0.2%. This is because the trained decision tree does not have preference and sacrifices 0.2% TP rate for increasing the TN rate by 1.5%. However, Algorithm 2 prefers improving TP to reducing FP, following our conservative design principle.

ZiSense is a general method that leverages distinct RSSI sequence patterns to recognize ZigBee. When applying to other environments, knowledge of the long-term existences of interference will help Algorithm 2 to improve accuracy. However, sensor nodes can encounter all the interference, especially for the system with mobile nodes [31]. As a general method, ZiSense takes all the common interference into account. Algorithm 2 is also scalable to handle the existence of other wireless technologies since the features are extracted from the standard specifications, hardware specification and etc, which will hold regardless of the application environments. When applying ZiSense to other platforms, ZiSense equipped with Algorithm 2 can be adopted by changing the parameter values accordingly to reduce the energy consumption caused by false wake-ups.

With the vigorous development of wireless technologies, emerging technologies may arise. When applying ZiSense in environments with new co-existing technologies, Algorithm 2 can still be adopted to identify ZigBee since the features of ZigBee signals will not change due to the appearance of other new technologies. But the false positive rate may increase if the underlying standards of new technologies own feature vectors similar to ZigBee. Then maybe new features need to be included for reducing the false positive rate. However, for now, Algorithm 2 is good enough to identify ZigBee from



(a) $T_{data} = 5min, n = 1, P = 0.8$ (b) $T_{data} = 5min, n = 5, P = 0.8$ (c) $T_{data} = 5min, n = 1, P = 0.2$ (d) $T_{data} = 5min, n = 5, P = 0.2$

Figure 12. The difference of duty cycle (ΔDC) between BOX-MAC-2 and ZiSense with different false positive rate and false negative rate. The traffic load is n packets in T_{data} time. The interference occurs with probability P. (a) low data rate, strong interference, (b) high data rate, strong interference, (c) low data rate, weak interference and (d) high data rate, weak interference. The dash line indicates the position where ΔDC$= 0$.

the common wireless technologies operated on 2.4GHz.

7 Performance Analysis

In this section, we analyze the performance of ZiSense with different settings (traffic load, interference situation, etc.). We also illustrate the influence of identification accuracy on the performance. We denote FN rate as p, FP rate as q. Then the TP rate is $1-p$ and the TN rate is $1-q$. The goal of all the rendezvous mechanisms in the low duty-cycled MAC protocols is to reduce the FP rate and FN rate, which correspondingly increase the TP rate and TN rate.

Suppose the data rate from the sender to the receiver is n/T_{data}, i.e., n packets in T_{data} time. We model the duty cycle of the sender-initiated protocols during the time slot of T_{data}. In each slot, the radio-on time consists of the following parts:

- *detection time (T_{detect})*: the time used at the receiver for detecting possible transmissions.
- *false wake-up time (T_{false})*: the time used at the receiver due to the false wake-ups caused by FP.
- *receiving time (T_{recv})*: the time used at the receiver for receiving packets.
- *transmission time (T_{trans})*: the time used at the sender for transmissions.

Denote the detection interval as T_0, the expected detection time is:

$$T_{detect} = \frac{T_{data}}{T_0} \cdot T_d \tag{8}$$

where T_d is the channel assessment time. If a node decides to stay awake, it keeps the radio on for T_w time. Suppose the probability that interference occurs when the node performs detection is P. Then the total false wake-up time in T_{data} is:

$$T_{false} = \frac{T_{data}}{T_0} \cdot P \cdot q \cdot T_w \tag{9}$$

The expected transmission time in T_{data} is

$$T_{trans} = (\frac{T_0}{2} + (\frac{1}{1-p} - 1) \cdot T_0) \cdot n \tag{10}$$

where $(\frac{1}{1-p} - 1) \cdot T_0$ is the retransmission cost. For simplicity, we assume that if ZigBee is detected, the node can receive the packet. Hence, the above retransmission cost only includes the retransmissions caused by detection errors without the retransmissions due to the poor link quality. Nevertheless, similar analysis can be applied to such kind of retransmissions.

The receiving cost for n packets at the receiver is:

$$T_{recv} = n \cdot T_{rx} \tag{11}$$

where T_{rx} is the cost for receiving one packet. Based on above equations, the duty cycle can be represented as:

$$DC(p,q) = \frac{T_{trans} + T_{recv} + T_{detect} + T_{false}}{T_{data}} \tag{12}$$

In BoX-MAC-2, nodes wake up as long as high energy is detected no matter what the signal is. Hence, its FP rate is 1 and FN rate is 0. ZiSense relies on the RSSI sequence pattern to recognize ZigBee. Its FP rate is much smaller than BoX-MAC-2, while FN rate is non-zero. We investigate the energy reductions by ZiSense with different FP and PN rates. We define the difference of duty cycle between BoX-MAC-2 and ZiSense with different FP and PN rates as follows.

$$\Delta DC(p,q) = DC_{BoX-MAC-2} - DC(p,q) \tag{13}$$

Positive $\Delta DC(p,q)$ means ZiSense consumes less energy than BoX-MAC-2. Negative $\Delta DC(p,q)$ indicates ZiSense costs more energy than BoX-MAC-2. The optimal duty cycle is $DC_{Optimal} = DC(0,0)$, which avoids all the false wake-ups and correctly detects all the ZigBee transmissions.

In Fig. 12, we show the duty cycle reduction under (a) a low data rate network with relative strong interference, (b) a high data rate network with relative strong interference, (c) a low data rate network with weak interference and (d) a high data rate network with weak interference. In each figure, we also illustrate the curve $\Delta DC(p,q) = 0$. According to default system settings, we set $T_{rx} = 100ms$, $T_w = 100ms$, $T_d = 2.9ms$ and $T_0 = 2s$.

Several observations are found in Fig. 12. First, Algorithm 2 adopted in ZiSense provides a quite low FN rate and FP rate, $p = 0.025$, $q = 0.024$, leading to 3.86%, 3.79%, 0.93% and 0.87% duty cycle reductions which are 97.2%, 95.4%, 95.8% and 88.8% approximated to the optimal solution under the settings (a), (b), (c) and (d) respectively. Second, there is more room for improvement in the network with a low data rate or a high interference. This can also be seen from the area of the region with $\Delta DC(p,q) > 0$.

Third, FN rate has a relative higher impact on the improvement, compared to FP rate. This is because the cost of each FN is a retransmission which takes $2s$ while the cost of each FP is a false wake-up which takes $100ms$. This is also the reason that in ZiSense's design, we fill in all the missing features with the values that satisfy ZigBee features in order to reduce the probability of FN.

Since AEDP can only filter out the interference with RSSI smaller than the weakest effective links, it has $p = 0$, $0 \leq q \leq 1$. Only if the majority interference is weaker than link RSSI, q is close to 0 so that AEDP outperforms ZiSense. If nodes only wake up when a packet is received during the short sampling period by setting CCA mode 2 [20], they have $0 \leq p \leq 1$, $q = 0$. Only if the majority packets are received during the short sampling period, p is close to 0. Then this method works better than ZiSense.

8 Implementation

We target on TelosB [8] motes to implement ZiSense under TinyOS 2.1.2 [2]. In Zisense, a sender will take certain time to wait the potential ACK between two adjacent transmissions of preamble packets. The RSSI sampling duration of a receiver should be longer than ACK waiting period to avoid mishearing any ongoing transmissions. According to the measurement results in AEDP [28], the ACK waiting period is at least 2.8ms. In our implementation, we set the RSSI sampling duration to 2.9ms, as the same as AEDP [28]. To obtain enough samples in such a short time, we increase the SPI speed and simplify the interfaces to quickly fetch RSSI readings from the register. The sampling frequency is increased to 31.25KHz, i.e., 32μs/sample. Hence, the sampling window of 2.9ms provides a RSSI sequence of $W = 90$ RSSI readings. The RSSI.RSSI_VAL register in CC22420 always has valid RSSI value when reception has been enabled at least 8 symbol periods (128μs). Hence, our implementation does not change the hardware RSSI sampling and just increase the register reading rate, which does not incur much extra energy consumption. During RSSI sampling, ZiSense also enable interrupt to quickly response the event of packet receiving.

After RSSI sampling, ZiSense will take extra CPU processing time to identify whether ZigBee exists. Based on our measurement, the average extra CPU processing time is 491.03μs on TelosB mote. We keep the radio on during ZigBee identification as well. Otherwise, if ZigBee is detected, the radio need be turned on again. Turning on CC2420 will take at least 580μs with higher energy consumption than receive mode [24]. It also will incur certain reception delay. Adding RSSI sampling duration 2.9ms, the total active time in ZiSense is about 3.4ms for one detection. The extra 0.5ms active time indeed increases the baseline energy, but the system performance can still be significantly improved because ZiSense greatly reduces the unnecessary energy consumption caused by false wake-ups, as demonstrated in Section 9. We summary the implemented parameters in ZiSense, as shown in Table 5. The parameters are configured by standards, hardware specifications and the real measurements.

For comparison, we implement AEDP according to the descriptions in the paper [28]. We also optimize the ACK waiting period from 10ms to 2.8ms, for Box-MAC-2. The implementation of ZiSense takes more space to store the RSSI sequence and algorithm codes. It consumes extra 1058 bytes RAM and 6344 bytes ROM, 32.2% and 23.1% more than the default implementation. However, the extra consumption is usually affordable for most of the sensor application programs on TelosB motes.

Table 5. Summary of parameters in ZiSense

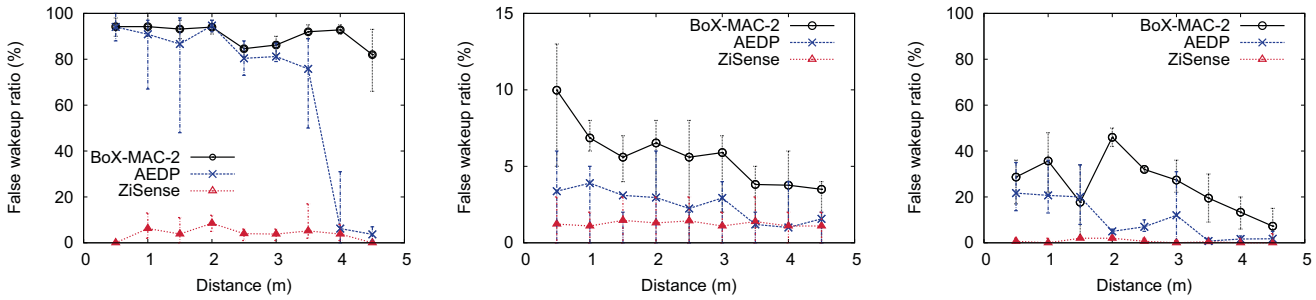
Notation	Value	Description
T_{ACK}	2.8ms	ACK delay [28]
MPI_{valid}	2.8ms, 192μs	Valid ZigBee *MPI* of unicast or broadcast [28] [20]
D_s	2.9ms	Channel sampling duration, longer than the maximum MPI_{valid}
T_{max}	4256μs	Maximum ZigBee on-air time, specified by the standard [17]
T_{min}	576μs	Minimum ZigBee on-air time, specified by the standard [17]
$PAPR_{ZigBee}$	1.3	Maximum *PAPR* of a valid ZigBee segment
δ	64μs	Error threshold for deciding the same length of time
ε	1dBm	Error threshold for deciding the same average RSSI
th_d	3dBm	RSSI threshold to detect the start and end points of a segment
th_n	-100dBm	RSSI of the minimum possible noise floor [20]
T_{rx}	100ms	Active time after receiving a packet, system default parameter
T_w	100ms	Active time after deciding to wake up, system default parameter

9 Evaluation

We evaluate the performance of ZiSense from the following aspects. First, we conduct the experiments with different interference sources to illustrate the effectiveness to mitigate false wake-up. We also study the impacts of link signal strengths on false wake-up mitigation. Second, we compare the total energy consumption among several protocols under different interference environments, to show our performance gain under controlled environments. Then we integrate ZiSense with CTP, and evaluate the energy, link and routing performance in a real-world data collection application. Since the energy consumed on the radio usually dominates other sources, we use the radio duty cycle as the aggregated energy indicator of the total energy consumption in our experiments.

9.1 False Wake-up

ZiSense should be able to solve the false wake-up problem. To verify this, we compare ZiSense with the CCA based mechanisms, i.e., AEDP and Box-MAC-2, under various controlled interference situations. We conduct the exper-



(a) WiFi 802.11b/g/n (b) Bluetooth (c) Microwave oven

Figure 13. The comparison of false wake-up ratio among ZiSense, BoX-MAC-2 and AEDP under the interference environments

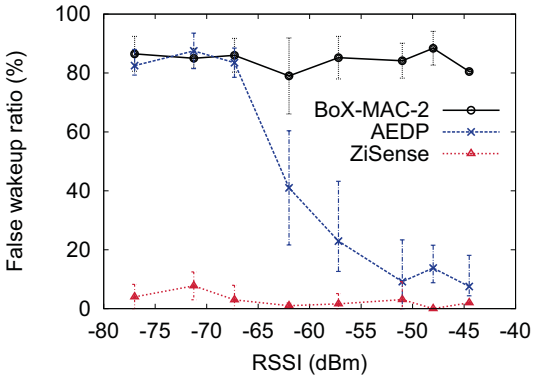


Figure 14. Impacts of link strength on false wake-up ratio

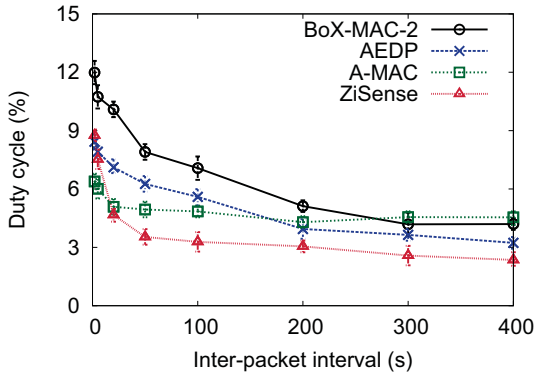


Figure 15. Impacts of data rate on duty cycle

iments in an empty room with maximum internal distance of 4.5m. We use LanTraffic V2 [3] software on two laptops to generate the WiFi signal. One laptop continuously transmits UDP data to the other laptop via a 802.11b/g/n AP at 5 Mbps data rate. For Bluetooth interference, we use a Bluetooth headset to listen to music on an iPhone 5 to generate the Bluetooth signal and configure two smartphones transmit an image file with size of 1M Bytes every 5 minutes. We use a Haier MJ-1870M1 microwave oven to heat a bowl of water to build the microwave interference. Then, we put a pair of sensor nodes at different distances from the interference source to vary the interference strength. The receiver checks the channel signal every 512ms. The sender transmits a packet every 10s. The RSSI of the ZigBee signal over the link is about -40dBm. The channel is set to 22.

The distributions of the false wake-up ratios are shown in Fig. 13. For different interference sources and interference strengths, ZiSense keeps a low false wake-up ratio. AEDP performs better than the default Box-MAC. Comparing with AEDP, the false wake-up ratio is reduced by 88.9%, 49.4% and 96.3% on average under WiFi, Bluetooth and Microwave interference. The results verify the effectiveness of ZiSense for conquering false wake-up problem. The results also reveal that the influence of WiFi signal is more serious than Bluetooth and Microwave. The frequency hopping in Bluetooth and the Faraday cage of microwave oven mitigate the impact of the interference on a certain channel.

9.2 Impacts of Link Signal Strength

We further explore the impacts of link signal strength on the performance of conquering false wake-up problem. We deploy a sender and three receivers in an office. The sender broadcasts a packet every 10s. The three receivers run BoX-MAC-2, AEDP and ZiSense respectively. The receivers calculate the average packet RSSI as the link RSSI. We vary the distance between sender and receivers to get various link signal strengths. In each run, the receivers count the number of wake-ups without receiving packets as the number of false wake-ups. At each location, we conduct ten runs of experiment. We group the links into 8 buckets based on link RSSI. Each bucket adopts the average link RSSI as representative link signal strength.

The false wake-up ratios of different methods are presented in Fig. 14. BoX-MAC-2 presents a high false wake-up ratio. AEDP effectively reduces the false wake-ups when link signal strength is strong ($RSSI \in [-55,-40]$). However, when link signal strength is weak ($RSSI \in [-80,-65]$), AEDP has as many false wake-ups as that BoX-MAC-2 has. Between the weak and strong regions, a transition zone ($RSSI \in [-65,-55]$) exists. In the transition zone, AEDP can only avoid partial false wake-ups. ZiSense keeps a low false wake-up ratio under various link signal strengths. This is because ZiSense leverages the differentiable signal features to distinguish ZigBee and interference, which do not vary with the link signal strength.

9.3 Impacts of Data Rate

We also explore the impacts of data rate on the performance of different mechanisms. We deploy 7 nodes and collect data with CTP in an office environment. The wake-up period of all nodes is 512ms for all mechanisms. 6 nodes send packets to the sink periodically. We configure inter-packet intervals (IPI) from 2s to 400s. The signal strength of routing links keeps around -70dBm. We run A-MAC, BoX-MAC-2, AEDP and ZiSense sequentially. We perform 5 experiments for each mechanism. We calculate the average radio duty cycles of all nodes.

Fig. 15 presents the results. When data rate is low ($IPI \geq 50s$), the duty cycle of A-MAC keeps around 4.5%, but the duty cycle of BoX-MAC, AEDP and ZiSense increases slowly with the same trend along the data rate gets high. The main reason is that the periodical probe-transmission/signal-detection dominates whole energy consumption. The efficient synchronized transmission in A-MAC further mitigates the influence of small incremental packet transmission on duty cycle. Due to ZiSense successfully avoids false wake-up, its duty cycle is approximate 24.9% and 42.8% lower than AEDP and BoX-MAC, respectively. Since sending probe of A-MAC will take more energy than ZigBee identification of ZiSense [28] and the possible probe loss of A-MAC, the duty cycle of ZiSense is at least 28.5% less than A-MAC. When data rate is high ($IPI \leq 50s$), the duty cycle of sender-initiated mechanisms increases faster than A-MAC. The main reason is that the huge amount of packet sending dominates whole energy consumption and the synchronized transmission in A-MAC leads higher energy efficiency. Moreover, the duty cycle of ZiSense increases more quickly than AEDP and BoX-MAC. This is because when the data rate is high, a node has high probability to detect the packet transmission of others and keep awake unnecessarily. In the worst case ($IPI = 2s$), the duty cycle of ZiSense is 27.1% and 3.7% worse than A-MAC and AEDP, but it still 26.9% higher than BoX-MAC. Overall, ZiSense efficiently avoids the influence of the non-ZigBee interference under various data rates and keeps the duty cycle low.

9.4 Duty Cycles

Given the effectiveness of ZiSense of solving false wake-up problem, we want to explore the performance gains in terms of the duty cycle. We evaluate the duty cycle of different nodes under different environments. We measure the duty cycles of A-MAC, BoX-MAC-2, AEDP and ZiSense under different environments. The link signal strength keeps be around -70dBm. The wake-up period on the receiver is 512ms. The sender generates a data packet every 10s. We take records about the radio-on time and the total time on both sender and receiver. Then we calculate the average of duty cycles on the sender and the receiver as the achieved duty cycle.

First, we conduct experiments in a $100\times50m^2$ office. 6 WiFi APs are deployed and the nearest one is 3m away from our sensor nodes. Several Bluetooth wireless earphones, keyboards and mouse are operated from 2m to 10m. Table 6 shows that the average duty cycle of ZiSense is 4.21%, which is the lowest. Compared to BoX-MAC-2, AEDP and A-MAC, ZiSense reduces the duty cycle by 61.2%, 49.8% and 17.1% separately. These results show the benefit that ZiSense experiences less false wake-ups and idle listening caused by interference.

Table 6. The comparison of the duty cycle among ZiSense, BoX-MAC-2, AEDP and A-MAC under different network conditions

	BoX-MAC-2	AEDP	ZiSense
Clean environment	3.31%	3.32%	3.39%
Office environment	10.86%	8.38%	4.21%
Severe interference	21.80%	18.87%	5.14%

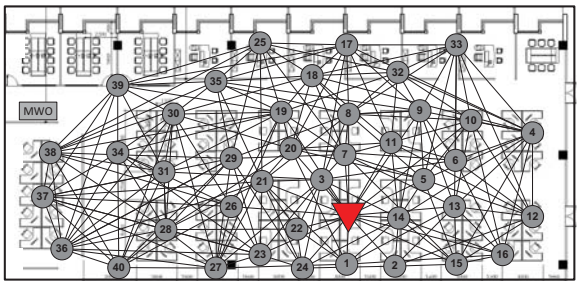


Figure 16. The topology of our indoor testbed

Then we use the UDP WiFi data transmission mentioned in Section 9.1 to build the severe interference environment (with the interference probability 0.9). We put the nodes at 3.5m away from the interference source. Table 6 shows the duty cycle of ZiSense is 5.14%, which is just 22.1% larger than the office environment. However, comparing with the office environment, the serious interference leads to 100.7%, 125.2% and 141.7% more energy consumption for BoX-MAC-2, AEDP and A-MAC separately. These results further illustrate the interference resilience of ZiSense.

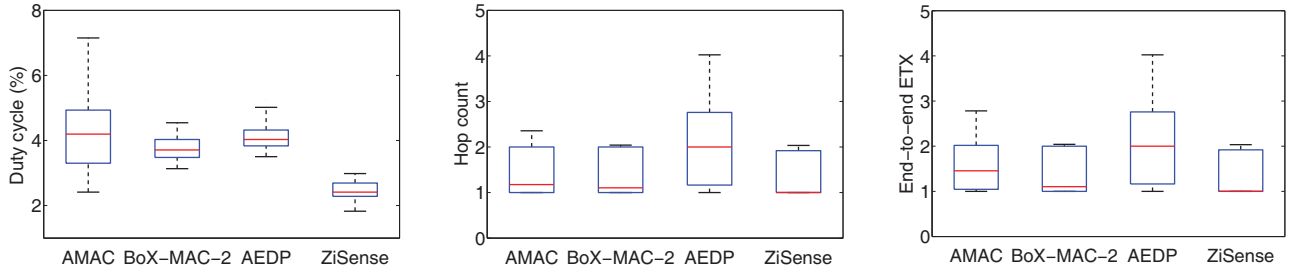
Finally, we move the pair of nodes in an outdoor playground without wireless interference. Table 6 plots the results. A-MAC achieves the lowest duty cycle, about 2.97%. This is because the sender with A-MAC predicts the time of probes to shorten the idle listening time. The duty cycle of BoX-MAC-2 and AEDP are comparable, about 3.31%. However, ZiSense results in 0.08% higher duty cycle than BoX-MAC-2 and AEDP. This is because ZiSense takes a longer time to process the RSSI sample sequence. Nevertheless, the baseline energy consumption of ZiSense is much smaller than its benefit.

9.5 Integration with CTP in the Real System

We investigate the energy, link and routing performance of ZiSense in a real indoor data collection deployment. We use the default CTP in TinyOS 2.1.2 as the routing scheme. We deploy 41 TelosB nodes in our $100\times50m^2$ office. The topology of our system, with a transmission power of 0dBm, is shown in Fig. 16. The sink node is labeled by a red triangle. The periodical wake-up interval is 2 seconds. Each node generates one data packet per 5 minutes. The communication

Table 7. Overall performance of different approaches. PDR, packet delivery ratio; RTX, retransmission count; ETX, expected transmission count which is the routing metric used in CTP.

Protocols	Duty cycle	PDR	RTX	Wake-ups per 5 min	Hop count	ETX
A-MAC	4.15%	99.26%	3.34	NA	1.33	1.44
BoX-MAC-2	3.74%	99.48%	0.10	61.61	1.42	1.43
AEDP	4.14%	99.65%	0.04	47.56	2.03	2.05
ZiSense	2.46%	99.79%	0.05	33.41	1.29	1.30



(a) Duty cycle (b) Hop count (c) End-to-end ETX

Figure 17. Box-plot comparison of A-MAC, BoX-MAC-2, AEDP and ZiSense, presenting the median, 25th percentile, 75th percentile and the range of duty cycle, hop count and end-to-end ETX

channel is set to 22, which overlaps with WiFi channel used by the office APs. Bluetooth interference comes from the wireless keyboards and mouse as well as Bluetooth headsets used by the staff in the office. A microwave oven operated during meal time is deployed at the location marked by the rectangle. We compare ZiSense with A-MAC, BoX-MAC-2 and AEDP. For each mechanism, we continuously collect the data for 24 hours, 4 days in total. We conduct all the experiments in workday to keep similar interference conditions.

Table 7 shows the average performance of duty cycle, packet delivery ratio (PDR), retransmission per packet (RTX), the number of wake-ups per 5 minutes, path hop count and routing ETX. The duty cycle of ZiSense is the lowest. The duty cycle of A-MAC is 68.7% higher than ZiSense. The reason is RTX of AMAC is 3.34, which is much larger than other protocols. The probe and packet loss is also verified, since the ETX is larger than hop count. The lower PDR of A-MAC reveals that single transmission of the probe is more vulnerable in co-existing environment, resulting in packet loss. However, in ZiSense, BoX-MAC-2 and AEDP, since the receiver might hear multiple preamble packets after waking up, the receiver has more chances to receive one correct packet. This is also why the RTX in these protocols is relatively small. All methods have similar PDR, illustrating that both AEDP and ZiSense will not decrease the PDR. The duty cycle reduction of ZiSense is achieved by reducing false wake-ups without sacrificing the delivery ratio of valid packets.

The duty cycle of BoX-MAC-2 is 52% higher than ZiSense since there are 84.4% more wake-ups for BoX-MAC-2 nodes. The duty cycle of AEDP is higher than BoX-MAC-2, but the number of wake-ups keeps low. Since the hop count of AEDP is larger than BoX-MAC-2, there are more packet relayed in AEDP. When the periodical wake-up interval is 2s, the relaying packets will consume more energy than wake-ups. The results infer that although AEDP could avoid the false wake-up by setting a high CCA threshold, it also increases the hop count of routing path and degrades the energy efficiency. In summary, ZiSense mitigates the energy inefficiency incurred by the interference. It also keeps the routing efficiency as much as possible.

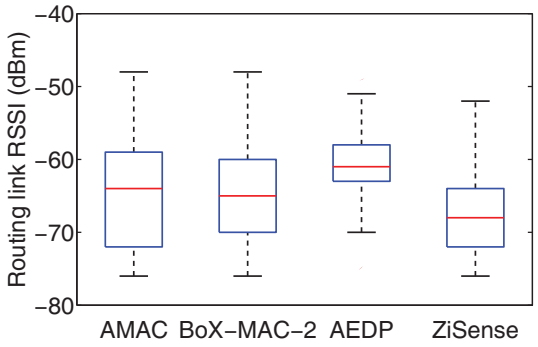


Figure 18. The distribution of the RSSI of routing links

Fig. 17 (a) presents the box-plot of duty cycles of different methods. AMAC has a high median duty cycle and a larger variation on the duty cycle, since the nodes affected by heavy interference have larger RTX which increases the duty cycle significantly.Fig. 17 (b) and (c) present the box-plots of the average path hop count and end-to-end ETX of all nodes.

Most of the nodes in ADEP have larger hop count than the other mechanisms. But the difference between ETX and hop count is small in AEDP, indicating the number of retransmissions is small. The reason is that using a higher CCA threshold in AEDP will filter some links with a low RSSI, as shown by the routing link RSSI distribution shown in Fig. 18. Fig. 18 shows the box-plots of the RSSI of the routing links in different protocols. The RSSI median of the routing links is around -67dBm in A-MAC, BoX-MAC-2 and ZiSense. Since AEDP increases the CCA threshold, the RSSI median of the routing links in AEDP is -61dBm, higher than that of others. As a result, AEDP selects shorter but more reliable links for routing, leading to a larger hop count and less retransmissions, as shown in Table 7.

10 Conclusion

We study the performance of different rendezvous mechanisms for low duty-cycled wireless sensor networks and investigate the fundamental problems in those protocols. We observe that both the signal strength based detection in LPL and packet probing based coordination in LPP are susceptible to interference. Under interference, the existing mechanisms are likely to waste energy, as a result of false wake-ups or idle listening. In this paper, we propose ZiSense, a new rendezvous mechanism in duty-cycled wireless sensor networks. Instead of relying on signal strength and probe packets, ZiSense leverages the featured patterns of ZigBee signals that are more resilient to interference. By avoiding unnecessary wake-ups, nodes in ZiSense reduce the energy consumption in noisy environment. We theoretically validate the performance gain of ZiSense, implement it in TinyOS, and evaluate its performance on TelosB motes with extensive experiments. The results show that, compared with existing rendezvous mechanisms, ZiSense significantly enhances the energy efficiency of sensor nodes.

11 Acknowledgments

The authors would like to thank the shepherd, Andreas Terzis, for his constructive feedback and valuable input. Thanks also to anonymous reviewers for reading this paper and giving valuable comments.

This work is supported in part by NSFC Major Program under grant No. 61190110, NSFC under grants No. 61202359, 61472217 and 61170213, National Science Fund for Excellent Young Scientist No. 61422207, and National Basic Research Program (973 program) under grant No. 2014CB347800.

12 References

[1] http://www.tinyos.net/tinyos-2.x/doc/html/tep111.html.

[2] http://www.tinyos.net/.

[3] Lantrafficv2. http://www.zti-telecom.com/EN/LanTrafficV2.html.

[4] B. Bloessl, S. Joerer, F. Mauroner, and F. Dressler. Low-cost interferer detection and classification using TelosB sensor motes. In *Proceedings of ACM MobiCom*, 2012.

[5] C. A. Boano, T. Voigt, C. Noda, K. Romer, and M. Zúñiga. Jamlab: Augmenting sensornet testbeds with realistic and controlled interference generation. In *Proceedings of ACM IPSN*, 2011.

[6] Z. Cao, Y. He, and Y. Liu. L^2: Lazy forwarding in low duty cycle wireless sensor networks. *IEEE/ACM Transactions on Networking*, PP(99):1–1, 2014.

[7] K. R. Chowdhury and I. F. Akyildiz. Interferer classification, channel selection and transmission adaptation for wireless sensor networks. In *Proceedings of IEEE ICC*, 2009.

[8] Crossbow Technology. TelosB mote platform. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf.

[9] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley-Interscience, 2001.

[10] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless. In *Proceedings of ACM SenSys*, 2010.

[11] Y. Gao, J. Niu, R. Zhou, and G. Xing. ZiFind: Exploiting cross-technology interference signatures for energy-efficient indoor localization. In *Proceedings of IEEE INFOCOM*, 2013.

[12] J.-H. Hauer, V. Handziski, and A. Wolisz. Experimental study of the impact of WLAN interference on IEEE 802.15. 4 body area networks. In *Proceedings of Springer-Verlag EWSN*, 2009.

[13] J.-H. Hauer, A. Willig, and A. Wolisz. Mitigating the effects of R-F interference through rssi-based error recovery. In *Proceedings of Springer-Verlag EWSN*, 2010.

[14] F. Hermans, O. Rensfelt, T. Voigt, E. Ngai, L.-Å. Nordén, and P. Gunningberg. SoNIC: classifying interference in 802.15. 4 sensor networks. In *Proceedings of ACM IPSN*, 2013.

[15] S. S. Hong and S. R. Katti. DOF: a local wireless information plane. In *Proceedings of ACM SIGCOMM*, 2011.

[16] J. Huang, G. Xing, G. Zhou, and R. Zhou. Beyond co-existence: Exploiting WiFi white space for Zigbee performance assurance. In *Proceedings of IEEE ICNP*, 2010.

[17] IEEE Computer Society. IEEE Standard 802.15.4. 2003.

[18] IEEE Computer Society. IEEE standard 802.15.1. 2005.

[19] IEEE Computer Society. IEEE standard 802.11. 2012.

[20] T. Instruments. CC2420 datasheet, 2007.

[21] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis. Surviving wi-fi interference in low power zigbee networks. In *Proceedings of ACM SenSys*, 2010.

[22] D. Moss and P. Levis. Box-macs: Exploiting physical and link layer boundaries in low-power networking. Technical report, Technical Report SING-08-00, Stanford University, 2008.

[23] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of ACM SenSys*, 2004.

[24] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Proceedings of ACM/IEEE IPSN*, 2005.

[25] S. Rayanchu, A. Patro, and S. Banerjee. Airshark: detecting non-WiFi RF devices using commodity WiFi hardware. In *Proceedings of ACM IMC*, 2011.

[26] S. Rayanchu, A. Patro, and S. Banerjee. Catching whales and minnows using WiFiNet: deconstructing non-WiFi interference using WiFi hardware. In *Proceedings of USENIX NSDI*, 2012.

[27] C. Schurgers. Systematic approach to peak-to-average power ratio in OFDM. In *International Symposium on Optical Science and Technology*, 2001.

[28] M. Sha, G. Hackmann, and C. Lu. Energy-efficient low power listening for wireless sensor networks in noisy environments. In *Proceedings of ACM IPSN*, 2013.

[29] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. Understanding the causes of packet delivery success and failure in dense wireless sensor networks. In *Technical report SING-06-00*, 2006.

[30] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. An empirical study of low-power wireless. *ACM Transactions on Sensor Networks*, 6(2):16, 2010.

[31] X. Wang, X. Lin, Q. Wang, and W. Luan. Mobility increases the connectivity of wireless networks. *IEEE/ACM Transactions on Networking*, 21(2):440–454, 2013.

[32] K. Wu, H. Tan, H.-L. Ngan, Y. Liu, and L. M. Ni. Chip error pattern analysis in IEEE 802.15. 4. *IEEE Transactions on Mobile Computing*, 11(4):543–552, 2012.

[33] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma. ZiFi: Wireless lan discovery via zigbee interference signatures. In *Proceedings of ACM MobiCom*, 2010.